



UNIVERSITÀ DELLA CALABRIA

DIPARTIMENTO DI
**INGEGNERIA INFORMATICA,
MODELLISTICA, ELETTRONICA
E SISTEMISTICA**

DIMES

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea

**Un approccio Deep Learning per la
previsione aziendale tramite KPI e
classificazione di immagini**

Relatore:

Prof. Antonella Guzzo

Prof. Francesco Scarcello

Studente:

Emilio Casella

Mat. 204898

A handwritten signature in black ink, appearing to read "Emilio Casella".

ANNO ACCADEMICO 2020/2021

Indice

Indice	1
1 Business Intelligence	7
1.1 Premessa: Cosa sono i Big Data?	7
1.2 La catena del valore	9
1.3 Business Analytics e Business Intelligence	13
1.4 Analisi dei Dati	14
1.5 Architettura dei sistemi BI	15
1.5.1 Sistemi di Data Warehouse:	17
1.5.2 Processi OLAP	20
1.6 Metodologia CRISP-DM	23
2 Sistemi di BI: Principali Soluzioni Disponibili	25
2.1 Analysis Service: Uno strumento per diversi utilizzi	25
2.2 Premessa: Il linguaggio DAX	26
2.3 Il cuore dell'Analysis Service:	
Il motore Vertipaq	27
2.3.1 Algoritmo 1: Codifica dei dati	28
2.3.2 Algoritmo 2: Dictionary Encoding	30
2.3.3 Algoritmo 3: Run Length Encoding	31
2.3.4 Relazioni in Vertipaq	34
2.3.5 Ottimizzazioni in Vertipaq:	
I concetti di materializzazione e aggregazione dei dati . .	35
2.4 Azure Analysis Services	37

<i>INDICE</i>	2
---------------	---

2.4.1 Premessa: Piattaforma PaaS	37
2.4.2 Modello di Sicurezza in Azure Analysis Service	38
2.4.3 Elaborazione di un modello	38
2.5 Power BI	39
2.5.1 Architettura	41
2.5.2 KPIs	42
3 L’Intelligenza Artificiale nel mondo del Business	45
3.1 Visualizzazione dei dati: perchè è importante?	45
3.2 Image Recognition: la nuova frontiera dell’IA	46
3.3 Che cos’è un’immagine digitale?	47
3.4 Ingegnerizzazione delle Immagini	49
3.4.1 Image Preprocessing	49
3.4.2 Image Analysis	49
3.4.3 Image Understanding	50
4 Tecniche di classificazione delle immagini	51
4.1 Algoritmi di Machine Learning	51
4.1.1 AdaBoost	52
4.1.2 Support Vector Machine	55
4.1.3 K-Nearest Neighbors	58
4.1.4 Random Forest	62
4.1.5 Reti Neurali	64
4.2 Algoritmi di Deep Learning	68
4.2.1 Reti Convoluzionali	69
4.2.2 Reti CNN & Image Recognition	69
4.2.3 Architettura CNN	69
4.2.4 LeNet-5	73
4.3 Funzione di Loss	74
4.4 Metriche di classificazione utilizzate	75
4.4.1 Matrice di Confusione	75
4.4.2 Precisione, Richiamo ed F1-Score	77

4.4.3 Curva di ROC	78
5 Business Forecasting	79
5.1 Metodologia sperimentale adottata	79
5.2 Creazione del Dataset	82
5.2.1 Aggregazione di KPI e preprocessing dei dati	85
5.2.2 Creazione delle immagini	92
5.2.3 Aggiunta vista andamento prezzi e consumi	97
5.2.4 Divisione del dataset	99
5.2.5 Etichettatura	107
5.2.6 Preprocessing per immagini e bilanciamento del dataset .	112
5.3 Classificazione	113
5.3.1 Tuning dei Classificatori	113
5.3.2 Tuning LeNet-5	116
5.3.3 Fase di training	120
5.3.4 Analisi dei risultati	121
5.3.5 Previsioni sull'anno 2020	127
5.4 Anomaly Detection	131
6 Conclusioni e scenari di sviluppo futuri	133
Bibliografia	136

Cos'altro dire?

Ne avrò sbagliate tante,

quello che so è che adesso scrivo giù le rime

e son più fitte di quelle di Dante.

(Bassi Maestro)

Introduzione

Durante il mio percorso di studi ho avuto modo di collaborare con EVO-BI s.r.l, Spin-Off Accademico dell' Università della Calabria.

L'azienda nasce dall'incontro tra il mondo della consulenza e la ricerca universitaria, con l'obiettivo di portare nelle piccole e medie imprese un sistema integrato di Business Intelligence, che sia agile potente ed efficace.

EVO-BI è un sistema immediato e plug-and-play, in grado di fornire al management indicatori chiave (KPI) ed avvisi intelligenti. La piattaforma è rivolta sia all'imprenditore che vuole iniziare a gestire la propria azienda secondo criteri manageriali, sia al manager più esigente che necessita dei più sofisticati sistemi di controllo direzionale. EVO-BI offre funzionalità di Business Intelligence evolute e subito disponibili per le PMI, ricavando informazioni di immediato valore che consentono all'impresa di prendere decisioni consapevoli e strategiche. Il processo di analisi e comprensione dei dati è dinamico, interattivo e soprattutto immediatamente comprensibile.

Tramite algoritmi di intelligenza artificiale, si cercano di individuare trend, outlier e più in generale importanti segnali da porre all'attenzione del management. In virtù di questo, nasce il mio lavoro di tesi.

Il mio ruolo all'interno del team di sviluppo mi ha permesso, dopo una prima fase di analisi interpretazione dei dati estrapolati dalla piattaforma, di creare modelli di classificazione e previsione, tramite le conoscenze acquisite durante il mio percorso di studi. Tutto questo, ai fini di unire un approccio reportistico, tipico della Business Intelligence, ad una nuova interpretazione dei dati tramite algoritmi specifici di IA.

L'idea chiave consiste nello sfruttare le immagini, ottenute attraverso gli indicatori KPI ed inerenti ad un cliente, che insieme all'utilizzo di tecniche ben note di machine learning e image recognition, dotate di algoritmi ben rodati e performanti, permettono di costruire modelli in grado effettuare analisi e previsioni aziendali.

L'elaborato parte da una overview sulla Business Intelligence, capitolo 1, analizzando i meccanismi delle principali soluzioni presenti sul mercato, relative al mondo Microsoft, capitolo 2. Il capitolo 3 pone l'accento sulla trasformazione dei dati in immagini che, tramite lo sfruttamento dell'intelligenza artificiale e di tecniche di image recognition, consentono di utilizzare in maniera intelligente tutti i dati a disposizione. Il capitolo 4, espone tecniche di classificazione e metriche, provenienti dal machine & deep learning, applicabili alle immagini. Il capitolo 5, elenca le varie fasi sperimentali. Dopo aver effettuato opportune operazioni sul dataset in esame, ricavato a partire dai KPI aziendali ed etichettato con una funzione euristica, si passa alla fase di classificazione. Vengono prodotti diversi modelli, a seconda del tipo di classificatore utilizzato, scegliendo il migliore sulla base delle metriche adottate. Una volta fatto ciò, si passerà ad una fase di previsione dell'andamento su un insieme di dati, nella stessa forma di quello utilizzato per la fase precedente ma non etichettato. Infine, si eseguirà uno studio del comportamento del modello scelto anche in fase di anomaly detection.

Capitolo 1

Business Intelligence

La Business Intelligence descrive una serie di concetti e metodi per migliorare il processo decisionale aziendale tramite sistemi informatici di supporto basati sui fatti, garantendo una visione della situazione precedente, attuale e futura dei clienti. Tutto ciò consente di migliorare le prestazioni aziendali, creando un contesto adatto a prendere decisioni utili al raggiungimento dei propri obiettivi. Le tecniche riguardanti la BI, devono utilizzare sia i dati strutturati sia quelli non strutturati per prendere una decisione accurata.

1.1 Premessa: Cosa sono i Big Data?

Big data è un termine che descrive un grande volume di dati, strutturati e non, che inonda le aziende ogni giorno. Ma non è la quantità di dati ad essere importante: ciò che conta veramente è quello che l'azienda fa con i dati. I big data, per contare qualcosa, devono essere analizzati andando alla ricerca di informazioni di valore, che portino a decisioni aziendali migliori e a mosse strategiche di business. Le informazioni trasmesse all'interno di una qualsiasi organizzazione, sono direttamente proporzionali all'incremento di complessità

della stessa.

Il primo fattore che entra in gioco è un incremento significativo del volume dei dati, che non possono essere inseriti all'interno di tabelle Sql e essere gestiti dal modello relazionale. Le raccolte dati sono state estese a nuove fonti che non provengono necessariamente dalle operazioni interne dell'azienda, ma che derivano da Internet, come testi, pagine Web, immagini e video, etc.[31].

Il secondo fattore che entra i gioco è la velocità. Grazie alle moderne infrastrutture di rete, che permettono lo scambio delle informazioni a velocità sempre più elevate, è stato possibile implementare dei servizi in cui un client riceve da un server grosse quantità di dati in tempo reale: si tratta dei cosiddetti servizi di streaming. La gestione di questi dati avviene secondo modalità totalmente diverse, rispetto a quelle dei dati offline, utilizzando un tipo di architettura in cui i dati non vengono archiviati ma processati attraverso l'elaborazione distribuita, necessaria anche quando i set sono relativamente piccoli[31].

Il terzo fattore che entra in gioco è la variabilità, rappresentando possibili cambiamenti nella velocità del flusso, nel formato, o nel volume, all'interno di un sotto-insieme di dati, che può portare alla necessità di ristrutturare completamente tutte le metodologie adottate. Per ottenere una gestione automatizzata ed efficiente, si utilizzano le funzionalità di ridimensionamento dinamico del cloud computing , poichè quest'ultimo consente maggiore efficienza[34].

Il quarto fattore determinante è il valore attribuito ad un determinato dato. La trattazione dei dati riveste, quindi, un ruolo determinante ai fini di un vantaggio competitivo aziendale. Vincolarsi all'oggettività ed empiricità dei dati significa adottare un approccio data driven, permettendo di prendere decisioni su fatti oggettivi e non su sensazioni personali.

1.2 La catena del valore

Si approfondisce ora il quarto punto analizzato nella sezione precedente, utile a comprendere il ruolo chiave che riveste il dato. In generale, la catena del valore rappresenta tutte le attività del ciclo produttivo che intervengono nei passaggi di trasformazione, partendo dalla materia prima ed ottenendo il prodotto finito. In ognuno di questi passaggi, il manufatto, acquisisce valore, fino ad arrivare all'ultimo step: il prezzo di vendita. Al contrario della prospettiva tecnologica, la prospettiva manageriale dei big data si concentra sulle domande legate al valore, utilizzando il concetto di catena per identificare i fattori cardine per la creazione di vantaggio competitivo aziendale.[27]

La catena del valore inizia con il catturare informazioni in formato digitale, seguita da una fase di raccolta, comprendente trasmissione e validazione di più fonti di dati. La terza fase è l'analisi, che comporta la scoperta di pattern all'interno dei dati, mentre l'ultimo step riguarda lo scambio dei dati ad un utente finale (cliente, utente interno all'azienda o un consulente esterno). A differenza della maggior parte delle catene del valore, in questo caso i dati non vengono "consumati" dall'utente finale, ma possono essere riutilizzati e divenire in futuro parte di una tendenza storica: per questo si utilizza il termine scambio.[32]



Figura 1.1: Catena del valore

- **Generazione**

La generazione è suddivisa in due sotto-fasi: acquisizione e consenso. Il primo passo è acquisire i dati da una fonte interna, creata da un essere

umano o da un dispositivo elettronico. Per ottenere un vantaggio competitivo, le imprese hanno bisogno che i dati generati siano di volume elevato. Così facendo, infatti, si è in grado di effettuare delle analisi migliori, rispetto a quelle attuate su un campione. Le aziende che riescono ad ottenere il consenso alla cessione dei dati da parte di un pubblico numeroso di utenti, sono in grado di acquisire un vantaggio competitivo notevole. Infatti, si è in grado di effettuare campagne di digital marketing più efficaci, tramite algoritmi di machine learning, con maggiori ritorni economici rispetto ai competitor.[32]



Figura 1.2: Fasi Generazione

- **Raccolta**

I dati devono essere trasmessi dal punto di acquisizione al punto di archiviazione, tramite un'infrastruttura di rete. Infine, devono essere inseriti in un processo di aggregazione, con quelli provenienti da altre fonti, e convalidati ai fini di garantire la correttezza e l'integrità degli stessi. Si creano, dunque, due sottofasi: trasmissione e convalida. La capacità di connettere e trasmettere dati, tra dispositivi e strumenti di archiviazione, è una funzione essenziale all'interno della value chain e man mano che sono raccolti più dati, a intervalli più frequenti, anche l'importanza di convalidare quest'ultimi aumenta. Nella fase appena citata, convergono tutte le operazioni che permettono di trasformare numerosi dati grezzi in una collezione da utilizzare per le successive operazioni, inerenti al data mining e al machine learning. All'interno di queste operazioni figu-

rano, innanzitutto, la verifica della correttezza e dell'integrità dei dati, l'integrazione di dati provenienti da diverse fonti ed eventualmente la definizione delle metriche e delle misure che sono interessanti ai fini dell'analisi. Lo scopo finale è aggregare i dati all'interno dei data mart, un sottoinsieme logico e fisico dei dati iniziali, che permetterà successivamente di ridurre i tempi necessari all'acquisizione dei dati da parte degli analisti [26]. La crescita della potenza di elaborazione e della capacità di archiviazione degli strumenti a disposizione, correlati alla possibilità per le aziende di utilizzare a costi molto ridotti servizi di cloud, permettono di sfruttare risorse elevate con investimenti non incenti.



Figura 1.3: Fasi Raccolta

• Analytics

La fase di Analytics, che comprende anche la fase di processamento iniziale, è utilizzata per trasformare i dati provenienti da database relazionali, data mart o filesystem distribuiti, tramite algoritmi matematici per segmentare i dati e costruire collegamenti. Parliamo quindi di data mining. Attraverso tecniche apposite di machine learning ed intelligenza artificiale, si può condurre e progettare l'analisi. La capacità di generare algoritmi di "autoapprendimento" è un mezzo per accelerare lo sviluppo di tali imprese le quali, attraverso l'analisi, possono realizzare business insight e presentazioni molto più fruibili [26]. Un buon algoritmo di questo tipo deve rispondere a quattro tipologie di analisi: descrittiva (identifica i

dati storici), diagnostica (capire le cause di una problematica), predittiva (cosa succederà) e prescrittiva (strategia di azione futura).

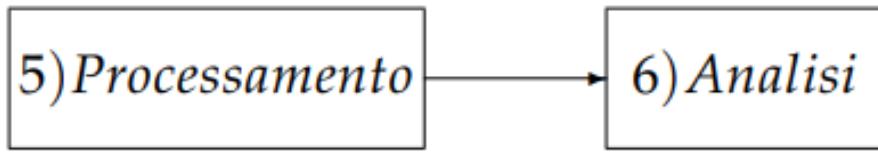


Figura 1.4: Fasi Analytics

- **Scambio**

In questo ultimo step, abbiamo due sottofasi: packaging e vendita oppure uso interno del tutto. Il packaging consiste nell'impacchettamento dei dati in diverse modalità, come la visualizzazione grafica o la reportistica, al fine di essere fruibili per gli utenti di business, che devono prendere decisioni inerenti le attività di marketing da effettuare.

La commercializzazione, invece, rappresenta il momento in cui il valore cumulativo, derivante dalle fasi precedenti, è maggiormente realizzato.[26]

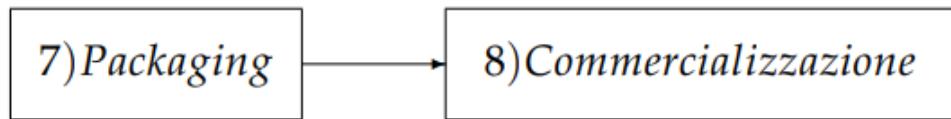


Figura 1.5: Fasi Scambio

1.3 Business Analytics e Business Intelligence

La necessità di coordinamento e compattezza, la frequenza ed i contenuti, rispecchiano la struttura del complesso organizzativo, suddiviso mediante differenti tipologie in chiave funzionale o divisionale. E' bene precisare che, in relazione alle interdipendenze che si vogliono favorire e a seconda del settore di contesto in cui opera l'azienda, i processi precedenti possono subire dei frazionamenti. Maggiore risulta la dinamicità dell'ambiente circostante e maggiore sarà la richiesta di flessibilità e coesione necessaria alla comunicazione delle informazioni, condizionando la tempestività di risposte e l'inerzia alla stocasticità dei cambiamenti esterni.[20]

Nel quadro descrittivo di queste strategie è possibile differenziare i diversi utilizzi dell'informazione elaborata dai dati. Mediante specifici sistemi e tecniche, è consentito proiettarsi sia ad una visione passata, sia ad un contesto temporale futuro con intento predittivo e prescrittivo [28]. La determinazione delle ramificazioni dell'uso dell'informazione consente a sua volta di distinguere, in modo analogo, le differenze tra la business intelligence e la business analytics. Quest'ultima cerca di anticipare eventi e situazioni, determinando nuovi modelli e correlazioni. Prima di effettuare aggregazioni, analizza nel dettaglio i dati, applica formule statistiche e successivamente trova correlazioni e modelli in grado di realizzare delle previsioni. Pertanto, utilizza diversi strumenti e tecniche come l'econometria, l'analisi di serie e l'analisi statistica, il rilevamento delle anomalie, l'ottimizzazione, la simulazione, l'analisi testuale e l'analisi quantitativa. La business intelligence si focalizza, invece, sulla valutazione delle serie storiche e dell'attimo presente, evidenziandone i trend e gli aspetti, così da realizzare benchmark attinenti[19].

1.4 Analisi dei Dati

Prima di affrontare qualsiasi discorso inerente alle piattaforme di BI, occorre conoscere le tipologie di dati che si possono avere a disposizione, i processi tramite i quali si ricavano e altre informazioni utili o significative all'analisi che si vuole intraprendere.

I dati possono provenire da svariate fonti, essere più o meno strutturati, attendibili, omogenei o disomogenei, completi o incompleti. L'insieme di queste caratteristiche determina la qualità generale del dato, insieme al grado di affidabilità.

L'acquisizione può provenire da diverse fonti come database di sistemi aziendali, log file di applicazioni, web service, documenti e file di testo, etc. Attualmente, i sistemi di BI sono focalizzati al trattamento di dati provenienti da sistemi di big data, ossia di grosse quantità di dati, difficilmente analizzabili con l'uso dei normali DBMS e provenienti da fonti eterogenee.

I dati, provenienti da più fonti, devono essere estratti, uniformati, normalizzati, combinati e convertiti nel formato più adatto alle successive interrogazioni. I dati grezzi, una volta acquisiti, devono essere ripuliti, uniformati, incrociati e riorganizzati, tramite operazioni che richiedono combinazioni di varie tecniche e relativi strumenti.

I dati ripuliti e uniformati sono solitamente inseriti e conservati in data warehouse, ovvero database con una struttura studiata appositamente per consentire analisi in real time di enormi quantità di dati. Quest'ultimi vengono caricati all'interno dei data warehouse parzialmente preelaborati, al fine di velocizzare le operazioni di analisi successive.

Una volta che si hanno a disposizione dei dati strutturati, si può andare alla ricerca di informazioni difficilmente ricavabili, in tempi ragionevoli, direttamente dai dati grezzi. Il tutto deve essere possibile in modo semplice, intuitivo e interattivo, tramite opportuni tools di visualizzazione.

Le informazioni estratte dai dati grezzi, sono spesso utilizzate al fine di monitorare alcuni aspetti dello scenario di applicazione e di segnalare tempestivamente anomalie nei valori dei dati, tramite l'impostazione di soglie d'allarme. All'avvicinarsi, dei valori rilevati, a tali soglie, può essere automatizzato l'invio di comunicazioni a persone chiave, le quali attuaranno le opportune azioni correttive.

Infine, le informazioni estratte vengono utilizzate per supportare le decisioni di chi occupa ruoli direzionali (sistemi per il supporto alle decisioni), ai fini di ridurre i costi, aumentare l'efficienza e identificare nuove opportunità di business. [29]

1.5 Arichitettura dei sistemi BI

L'architettura dei sistemi di Business Intelligence è costituita da tre livelli fondamentali:

- **Sistemi alimentanti:** Primo livello composto dai sistemi che producono dati elementari.
- **Sistemi di Data Warehouse:** Livello composto da sistemi di data management progettati per abilitare e supportare le attività di business intelligence (BI), in particolare gli analytics. I data warehouse servono esclusivamente a eseguire query e analisi, che spesso contengono grandi quantità di dati storici. I dati all'interno di una struttura di questo tipo sono generalmente derivati da una vasta gamma di origini, come i file di registro dell'applicazione e le applicazioni di transazione.
- **Sistemi di Business Intelligence:** Sistemi per l'accesso ai dati e la produzione di informazioni che costituiscono il livello finale. Nel primo livello, occorre raccogliere e integrare i dati presenti presso le diverse sorgenti primarie e secondarie, eterogenee per provenienza e tipologia. Tutto

ciò si basa prevalentemente sui dati dei sistemi operazionali (ERP,CRM, SCM, etc.), includendo in alcuni casi anche documenti non strutturati.

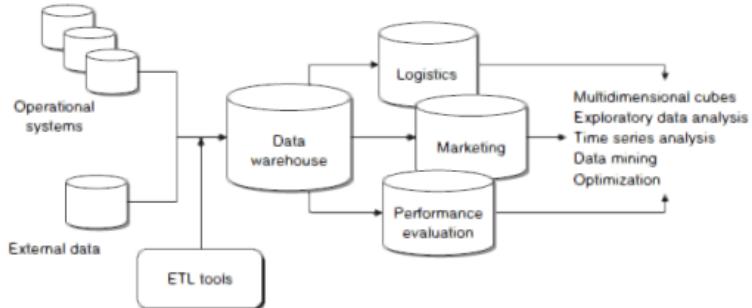


Figura 1.6: Architettura Sistemi BI

In generale, fanno parte del primo livello tutte le informazioni provenienti sia dall'interno che dall'esterno dell'impresa. Successivamente, questi dati sono sottoposti ad un processo di estrazione, pulizia e trasformazione, denominato ETL (extract, transform, load), prima di essere caricati nel data warehouse.

Le procedure di popolamento possono raggiungere elevati livelli di complessità, in relazione alle discrepanze esistenti tra le sorgenti, al loro livello di correttezza e al livello di precisione rappresentativa nel tempo che si desidera mantenere. Infine, si passa ai sistemi di BI che consentono la ricerca intelligente di dati e la produzione e analisi in “tempo reale”, estrapolando le informazioni di cui si necessita nel rispetto dei propri tempi decisionali; permettendo di osservare un fenomeno aziendale o analizzare un problema emerso. Come anticipato anche precedentemente, svolge un ruolo cruciale il processo ETL, effettuato tramite software specifici [18]:

- **Estrazione:** divisione dei dati in base alle fonti (interne o esterne).
- **Pulitura e Trasformazione:** si cerca di migliorare la qualità dei dati estratti dalle diverse fonti, mediante la correzione di eventuali inconsistenze, inesattezze e carenze. Nella fase di pulitura vengono applicate

regole automatiche predefinite per la correzione di eventuali errori. La fase di trasformazione, invece, prevede ulteriori conversioni dei dati che ne garantiscono l'omogeneità, rispetto all'integrazione delle diverse fonti. Vengono anche eseguiti calcoli di ai fini aggregativi delle informazioni, in modo tale da ottenere le sintesi richieste per svolgere le analisi cui il data warehouse è rivolto.

- **Refresh e Update:** i dati vengono completamente riscritti all'interno del data warehouse. Questa tecnica di caricamento viene utilizzata nella fase di inizializzazione, o meglio quando si procede all'estrazione di tutti i dati storici, aggiungendo solo i nuovi e aggiornando il tutto. Questo processo viene ripetuto periodicamente nel tempo, al fine di preservare correttezza e consistenza dei dati da analizzare.

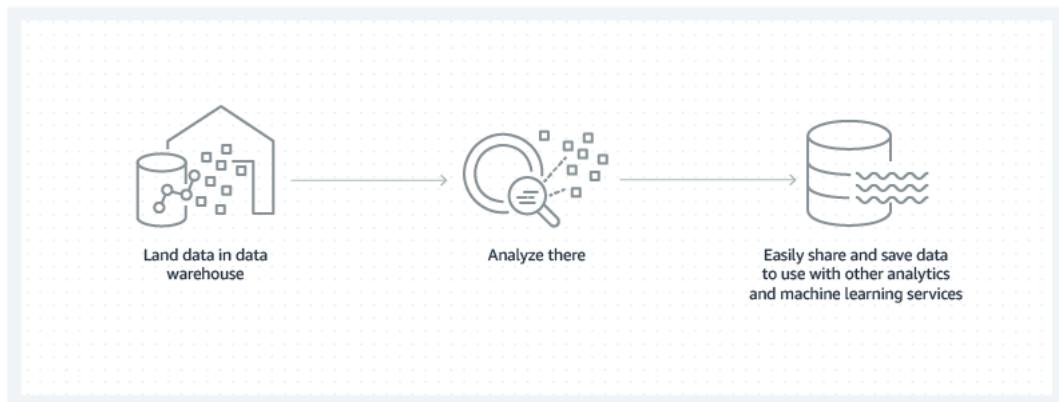


Figura 1.7: Data Warehouse

1.5.1 Sistemi di Data Warehouse:

Un vero e proprio spartiacque, nella moderna gestione dei dati, è derivato dalla nascita del linguaggio Sql. Negli anni è stato portato avanti il modello relazionale, all'interno di strumenti etichettati come Database Management System (DMBS). La struttura risultante viene denominata RDBMS. Questo modello

organizza i dati in una o più tabelle , o ”relazioni”, di colonne e righe, con una chiave univoca che identifica ciascuna di esse. Le righe sono anche chiamate record o tuple, mentre le colonne sono anche chiamate attributi.

In generale, ogni tabella o relazione rappresenta un ”tipo di entità”, le righe rappresentano istanze di quel tipo di entità e le colonne rappresentano i valori attribuiti a quell’istanza.[26]

Negli anni, le tecniche di gestione dei database relazionali hanno portato a dei processi di normalizzazione delle tabelle, con l’obiettivo di ridurre la ridondanza dei dati e migliorarne l’integrità[29].

Il modello non relazionale, spesso identificato anche con il termine NoSql, indica invece un distacco radicale rispetto a quello tradizionale. Le sue versioni più comuni sono il modello documentale e il modello a grafo. Il modello relazionale si è rivelato molto adatto per l’elaborazione parallela, l’elaborazione client-server e le GUI. All’interno delle strutture RDBMS, è bene citare i processi OLAP (On Line Analytical Processing), le quali attraverso ricerche all’interno dei database, sono finalizzate ad operazioni di analisi o più semplicemente alla restituzione di informazioni.

Le prime risposte tecnologiche, sono pervenute attraverso la nascita dei data warehouse, descritti come dei contenitori centralizzati di informazioni, in cui i dati confluiscono a partire dagli stessi RDBMS e da altre fonti. Si tratta di una collezione fisicamente separata dai sistemi operazionali, con le seguenti caratteristiche [17]:

- **Orientata agli oggetti** : possono analizzare dati su un particolare argomento o area funzionale (come le vendite).
- **Integrata**: i data warehouse creano coerenza tra diversi tipi di dati provenienti da origini disparate.
- **Non volatile**: una volta che i dati si trovano in un data warehouse, sono stabili e non cambiano.

- **Variabile con il tempo:** l'analisi del data warehouse esamina il cambiamento nel tempo.

Esistono due approcci fondamentali nella realizzazione di un data warehouse [17]:

- **Query-driven:**

un "mediatore" genera delle sottoquery per i vari DBMS eterogenei, mette insieme i risultati e risponde alla query originale.

- **Update-driven:**

l'informazione è integrata in anticipo, non causando interferenza tra query al DW e query ai singoli database.

Tutte queste informazioni, hanno portato alla creazione di 5 tipologie di architetture:

- **Data Mart indipendenti:**

un data mart è un DB strutturato in base all'argomento, una sorta di sezione del data warehouse. In questo caso, però, diversi data mart non sono collegati tra loro, magari poichè si riferiscono a compagnie diverse etc.

- **Bus:**

simile alla struttura precedente, forma un anello di congiunzione tra i data mart, costituendo un modello unico per tutti i dati.

- **Architettura hub e spoke:**

architettura che presenta livelli in cui diversi data mart sono aggregati, secondo determinate metriche.

- **Architettura centralizzata :**

architettura in cui non si trovano singoli data mart, ma esclusivamente trattata come repository centralizzata.

- **Architettura confederata :**

rappresenta un sistema integrato di diversi data warehouse e data mart, realizzato attraverso tecniche come querying distribuito, ontologie e interoperabilità dei metadata.

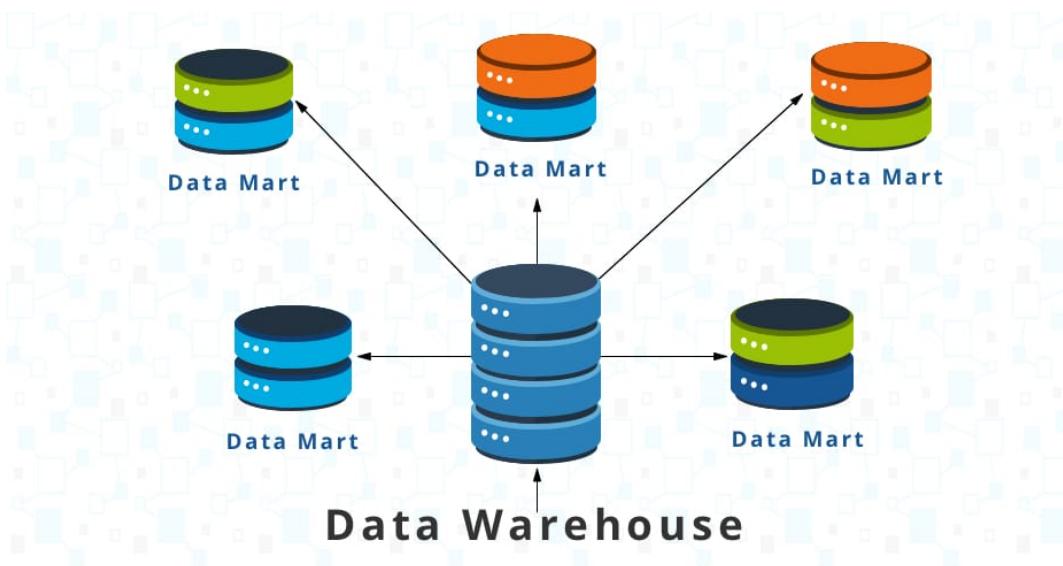


Figura 1.8: Data Warehouse 2

1.5.2 Processi OLAP

Gli strumenti che facilitano le operazioni di analisi, effettuate dai data mart, vengono definiti sistemi OLAP, analizzando grandi quantità di dati con software complesso. Esistono tre tipologie di server OLAP [26]:

- **ROLAP:**

i dati sono forniti direttamente dal data warehouse principale e memorizzati in vettori, con accesso di tipo posizionale. Il sistema alloca una cella per ogni possibile combinazione dei valori delle dimensioni, garantendo un accesso ad un determinato attributo in modo diretto e sulla base delle coordinate fornite.

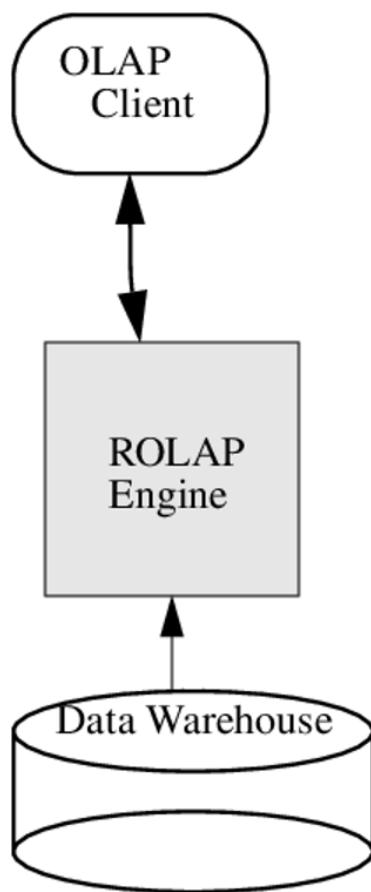


Figura 1.9: Architettura ROLAP

- **MOLAP:**

forniscono i dati a partire da database multidimensionali (MDDB), associati concettualmente a dei cubi, la cui dimensione è correlata a quelle del modello. I punti interni simboleggiano tutte le misure che si possono

effettuare, ma dato che solitamente il modello è rappresentato da più di tre dimensioni, avrebbe più senso utilizzare il termine di ipercubo.

In sistemi architettonici di questo tipo, il vantaggio è la velocità di accesso, mentre il limite è legato alla elevata complessità del cubo, correlata all'aumento di dimensioni e gerarchie.

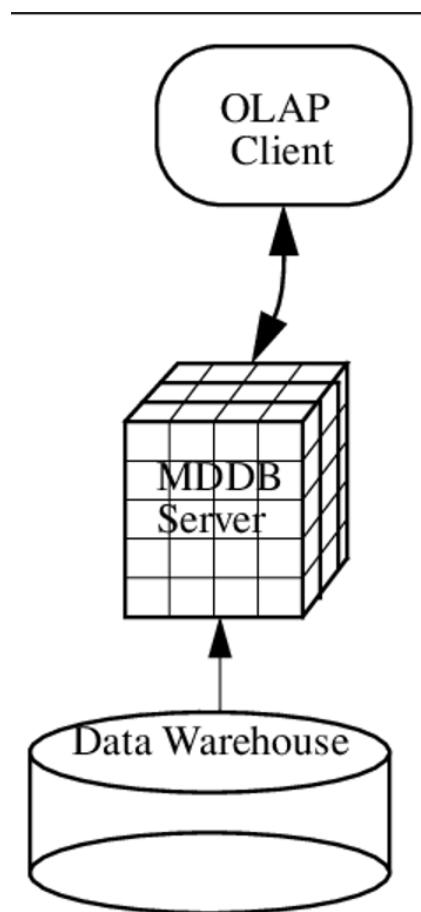


Figura 1.10: Architettura MOLAP

- **HOLAP:**

architetture ibride che combinano sistemi ROLAP e MOLAP.

1.6 Metodologia CRISP-DM

I dati conservati all'interno dei server OLAP sono ora pronti per essere utilizzati, per le operazioni di analytics. La scelta del migliore algoritmo rappresenta solo l'ultimo step conclusivo. Il tutto è chiarito dalla metodologia CRISP-DM [33]:

- **Comprendere il Business:** intuire gli obiettivi del progetto utili al cliente.
- **Comprendere i dati:** analizzare caratteristiche e qualità dei dati.
- **Preparare i dati:** selezione di record e attributi da utilizzare nel modello, effettuando eventuali trasformazioni.
- **Build del modello:** selezione del modello migliore, a seconda della tipologia di analisi da effettuare.
- **Valutazione del modello:** effettuare testing per capire la bontà e l'efficacia di ciò che si è prodotto.
- **Sviluppo:** rilascio del modello.

Le seguenti fasi hanno ispirato il processo sperimentale descritto nel seguito dell'elaborato permettendo, tramite la combinazione di tecniche di data mining e machine learning, di produrre un modello utile al caso oggetto di studio.

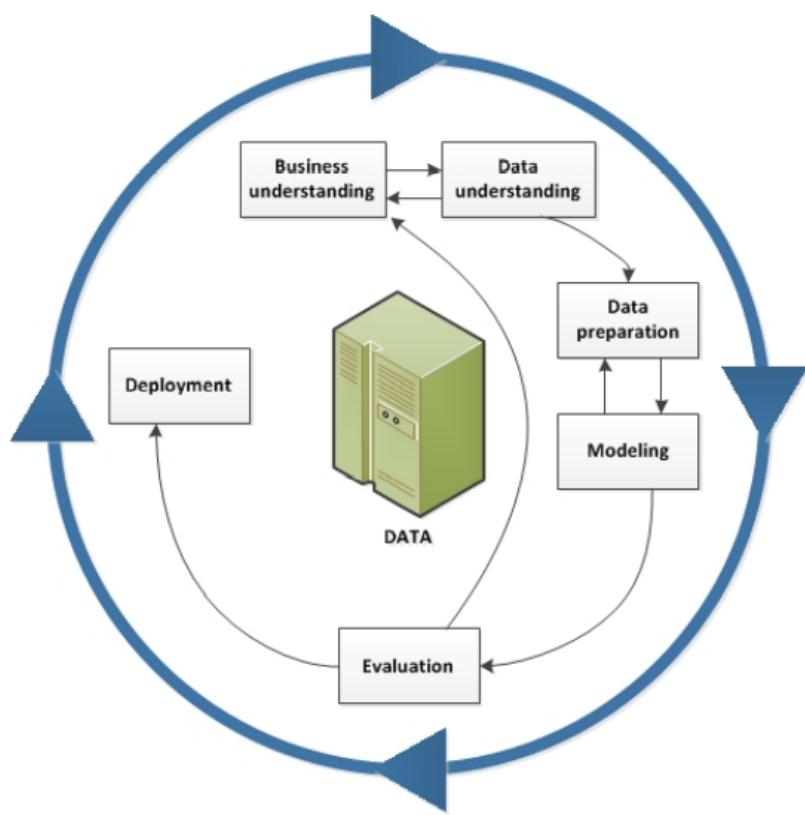


Figura 1.11: Modello CRISP-DM

Capitolo 2

Sistemi di BI: Principali Soluzioni Disponibili

Le principali piattaforme di supporto alla BI sono fornite da Microsoft, sempre all'avanguardia nel fornire soluzioni efficienti per la previsione aziendale.

2.1 Analysis Service: Uno strumento per diversi utilizzi

Analysis Service è un motore di dati analitici, sviluppato da Microsoft. E' utilizzato nel supporto decisionale e nell'analisi aziendale, tramite funzionalità del modello di dati semantico, utile per applicazioni di business intelligence, analisi dei dati e creazione di report.

L'idea rivoluzionaria è rappresentata dall'applicare questo servizio a varie piattaforme, con obiettivi finali diversi:

- Power BI
- Azure Analysis Service

Analysis Service consente molta più flessibilità arrivando ad essere, tramite l'astrazione, una sorta di livello semantico che rende meno difficile, per uno sviluppatore, soddisfare l'esigenze dell'utente finale.

2.2 Premessa: Il linguaggio DAX

Il linguaggio Dax è utilizzato per calcolare formule computazionali specifiche a seconda del modello di dati utilizzato, ad esempio dati tabellari.

Se un modello contiene molte tabelle, è altamente probabile che queste siano collegate da una relazione di tipo unidirezionale, bidirezionale o incrociata.

DAX offre funzioni per modellare, creare o analizzare i dati, raggruppati in funzioni di aggregazione, di conteggio, logiche, informative, di testo etc.

Il linguaggio è utilizzato in diversi prodotti della famiglia Microsoft, basati su tipologia tabulare, mantendendo la stessa base e implementando ulteriori funzioni specifiche a seconda del contesto di utilizzo del prodotto.

I modelli tabulari utilizzano due motori per elaborare una query [30]:

- **Il motore delle formule (FE):** Elabora la richiesta.
- **Il motore di archiviazione (SE):** Recupera i dati dal modello tabulare per rispondere alle richieste avanzate da FE.

Il motore FE può avere due implementazioni: DirectQuery, che inoltra le query direttamente all'origine dati per ogni richiesta, e VertiPaq che ospita una copia dei dati in memoria, aggiornandola periodicamente dalla fonte dei dati.

Proprio Vertipaq è alla base dell'Analysis Service.

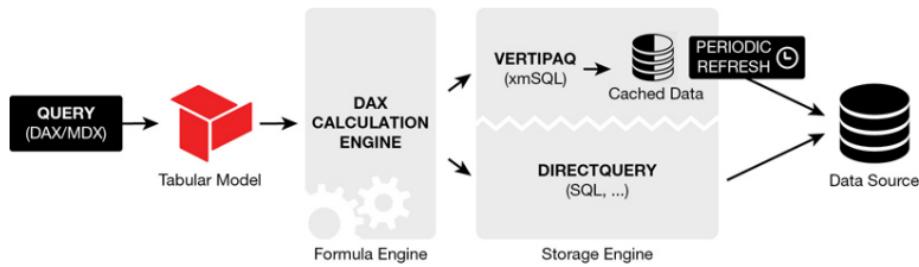


Figura 2.1: Esecuzione Query DAX

2.3 Il cuore dell'Analysis Service: Il motore Vertipaq

Il motore di archiviazione VertiPaq è l'unità di esecuzione nativa di basso livello del linguaggio DAX, archiviando una copia dei dati letti dalla sorgente in un formato compresso in memoria e basato su una struttura di database a colonne. Le query VertiPaq vengono espresse utilizzando un linguaggio pseudo-SQL interno chiamato xmSQL, non un linguaggio vero e proprio per query ma piuttosto una rappresentazione testuale di un motore di archiviazione per interrogazioni.

Le operazioni eseguite dal VertiPaq SE sono molto efficienti e possono scalare su più core, aumentando il suo parallelismo fino a un thread per ogni segmento di una tabella.

Un sistema di cache memorizza i risultati prodotti dal motore di archiviazione VertiPaq, restituendo in genere le ultime 512 query interne al database.

Quando quest'ultima componente riceve una query xmSQL identica a quella già presente nella cache, restituisce il corrispondente datacache senza effettuare alcuna scansione dei dati in memoria. Un'operazione di scansione eseguita

dal motore SE è in genere più veloce della scansione equivalente eseguita dal motore della FE, in quanto lo storage engine è ottimizzato per iterare su dati compressi.

Ricapitolando, il motore vertiPaq è alla base di tutte le versioni di Analysis Service tabulari e anche di Power BI. L'approccio chiave è salvare tutti i dati in memoria, tramite opportuni processi di compressione, rendendo molto efficiente l'esecuzione di calcoli computazionalmente complessi al giusto compromesso.

Il cuore di Vertipaq è composto da tre tipologie di algoritmi[23].

2.3.1 Algoritmo 1: Codifica dei dati

Il primo passo da compiere è la compressione di dati in input.

Si procede applicando un'operazione matematica ai dati numerici, con l'obiettivo di ridurre il numero di bit necessari per memorizzare ciascun valore. L'operazione inversa viene quindi eseguita quando i dati vengono letti da una query.

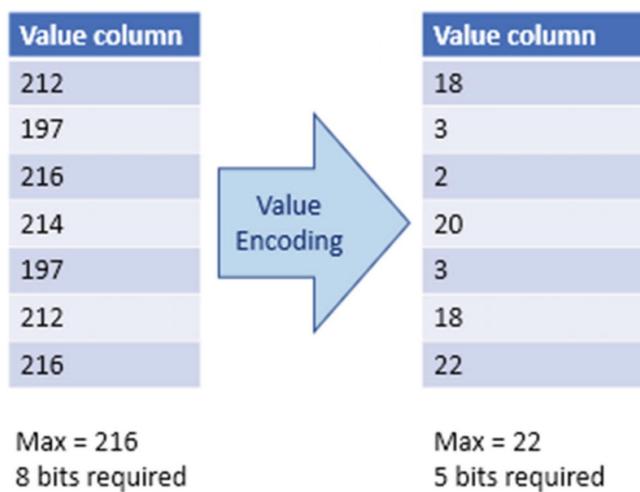


Figura 2.2: Esempio per Algoritmo 1

Ovviamente, i passi compiuti dal motore Vertipaq sono più complessi rispetto alla figura sovrastante.

Ad esempio, l'algoritmo può rilevare correlazioni matematiche intrinseche tra i valori di una colonna, modificando lo stato di archiviazione in base a queste informazioni. Il tutto può avvenire prima o dopo la fase di aggregazione dei dati.

La compressione dei dati è un passaggio fondamentale.

Diversi miliardi di righe non vengono ridotte in un unico passaggio. La tabella viene elaborata come una singola unità divisa in segmenti che contengono, ad esempio, 8 milioni di righe ciascuno. Quando un segmento è completamente letto, il motore inizia a comprimerlo nello stesso momento in cui legge quello successivo.

Vertipaq utilizza la segmentazione come base per applicare il parallelismo, utilizzando un core per segmento durante la scansione di una colonna. Più grande è il segmento, migliore è la compressione.

Tuttavia, aumentare la dimensione del segmento può avere effetti negativi, influenzando notevolmente il tempo di elaborazione: più grande è il segmento, più lenta è l'elaborazione.

Se, invece, i segmenti sono piccoli il parallelismo nella lettura della query aumenta. Anche se è vero che la scansione della colonna è più veloce perché più core possono farlo parallelamente, Vertipaq ha bisogno di più tempo alla fine della scansione per aggregare i risultati parziali calcolati dai diversi thread. Se una partizione è troppo piccola, il tempo necessario per gestire il cambio di attività e l'aggregazione finale è più del tempo necessario per scansionare i dati, con un impatto negativo sulle prestazioni complessive della query.

Ricapitolando, il partizionamento eccessivo di una tabella è un errore comune realizzato da principianti, ottenendo l'effetto opposto: creare troppe piccole partizioni che riducono le prestazioni ottenute [24].

2.3.2 Algoritmo 2: Dictionary Encoding

Il secondo algoritmo traspone una serie di parole in un dizionario, tenendo conto dell'indicizzazione.

Tramite questo processo, si procede ad una memorizzazione nel modello, sostituendo il valore attribuito, nel dizionario, a quello testuale ed ottenendo un buon grado di compressione.

Ci sono alcuni vantaggi nell'utilizzare questo tipo di codifica: tutte le colonne contengono solo valori interi, rendendo più semplice ottimizzare l'interno codice del motore; rendendo, inoltre, Vertipaq indipendente dalla tipologia del dato in esame.

Il numero di bit utilizzati per memorizzare un singolo valore è minimo, o meglio quello necessario per memorizzare una voce dell'indice. I dati così codificati, forniscono le stesse prestazioni in termini di velocità di scansione e di spazio di archiviazione.

L'unica differenza potrebbe essere nella dimensione del dizionario, che è tipicamente molto piccola rispetto alla dimensione della colonna originale stessa.

Il fattore principale per determinare la dimensione della colonna è il numero di valori distinti nella stessa, inteso come cardinalità. Minore è la cardinalità, minore è il numero di bit necessari per memorizzare un singolo valore.

Se una colonna è più piccola, non solo sarà possibile memorizzare più dati nella stessa quantità di RAM, ma sarà anche molto più veloce scansionarli ogni volta che il motore ha bisogno di aggregare i suoi valori in una espressione scritta in DAX.

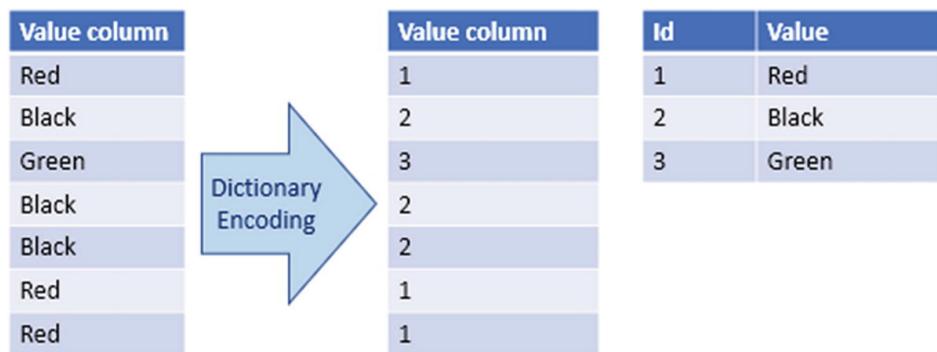


Figura 2.3: Implementazione Algoritmo 2: Dictionary o Hash encoding

2.3.3 Algoritmo 3: Run Length Encoding

L'ultimo algoritmo si pone il compito di rimuovere la quantità di dati ridondanti nel modello.

Viene creato un nuovo dizionario, contenente ogni parola con relativa occorrenza nelle righe, richiamando questo dizionario a runtime e utilizzandolo opportunamente.

In altre parole, Vertipaq evita la memorizzazione di valori che si ripetono, sostituendoli con una struttura leggermente più complessa che contiene il valore una sola volta, insieme al numero di righe contigue aventi lo stesso valore.

L'efficienza di RLE dipende fortemente dallo schema di ripetizione della colonna. Alcune hanno lo stesso valore ripetuto per molte righe, risultando in un ottimo rapporto di compressione. Altre, invece, hanno valori che mutano rapidamente, portando ad una compressione minore del dato.

E' bene porre l'accento sul fatto che l'ordinamento dei dati è estremamente importante per migliorare il rapporto di compressione di RLE.

Pertanto, trovarne uno ottimale è un passaggio cruciale.

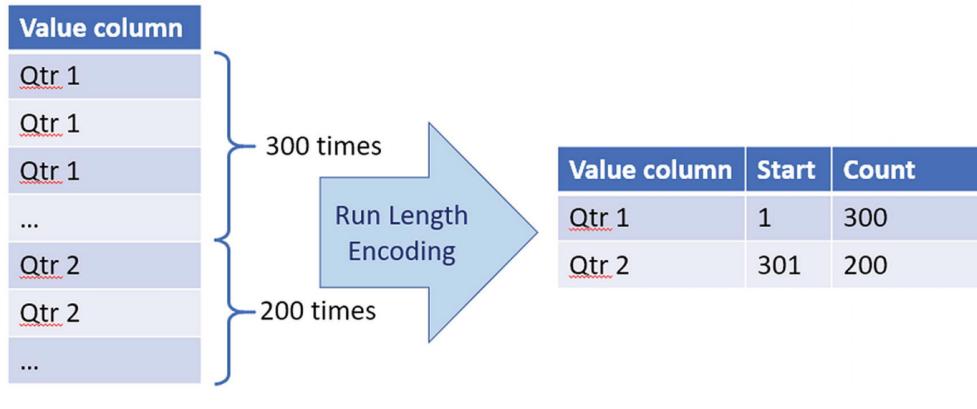


Figura 2.4: Implementazione Algoritmo 3

Occorre precisare che l'algoritmo utilizzato ottiene scarsi risultati su colonne in cui il contenuto cambia molto spesso, come la chiave primaria di una tabella. In questi casi, si salta la compressione RLE e si memorizza la colonna così com'è, ottendendo che l'archiviazione non supera mai la dimensione della colonna originale.

Come fa dunque l'algoritmo a capire se utilizzare la codifica del dizionario oppure utilizzare il valore così com'è?

In prima istanza, si leggono i valori di una riga presa come campione. Se il tipo di dati della colonna non è intero, si utilizza la codifica del dizionario, altrimenti si applicano alcune euristiche.

Ad esempio, se i numeri nella colonna aumentano in modo lineare, probabilmente sto scandendo una chiave primaria e bisogna lasciare il dato così com'è.

Se, invece, i numeri rientrano in un range di valori molto ampio con valori molto diversi tra loro, la codifica tramite un dizionario è la scelta migliore.

Una volta presa la decisione, si inizia a comprimere la colonna. Se durante la codifica l'algoritmo si accorge di aver intrapreso una scelta non corretta per una colonna, può procedere con una ricodifica della stessa.

Tuttavia, questa eventuale operazione incide molto in termini di costo, soprattutto per grandi moli di dati, rimarcando l'importanza di fornire in input dati

con un buon grado di pulizia.

E' bene specificare che per migliorare l'efficienza di RLE, nelle tabelle di grandi dimensioni, è importante determinare il miglior ordinamento dei dati possibili, ottimizzando lo spazio in memoria occupato dalle stesse.

Quando l'algoritmo legge una tabella, prova diversi tipi di ordinamento per migliorare la compressione. In un tabella con molte colonne, questa è un'operazione molto costosa. Si imposta, quindi, un limite superiore al tempo che può dedicare alla ricerca del miglior ordinamento possibile. L'impostazione predefinita può cambiare con diverse versioni del motore e del tipo di prodotto in uso.

D'altra parte, l'elaborazione richiederà molto più tempo perché il motore viene incaricato di provare tutti i possibili ordinamenti prima di effettuare una scelta. In genere, gli sviluppatori dovrebbero mettere prima le colonne con il minor numero possibile di valori univoci nell'ordinamento, poichè è altamente probabile che queste generino numerosi valori ripetuti.

Da tutte queste considerazioni, però, si può dedurre che la scelta dell'algoritmo di ordinamento corretto ha un reale impatto a livello di costo, quando si entra in relazione con un modello di dati davvero grande, nell'ordine di pochi miliardi di righe. In caso contrario, il beneficio ottenuto da queste ottimizzazioni estreme è limitato.

Una volta effettuate le operazioni di compressione, si completa l'elaborazione costruendo colonne, tabelle, gerarchie e relazioni. Le gerarchie e le relazioni sono strutture dati aggiuntive necessarie a Vertipaq per eseguire query, mentre colonne calcolate e tabelle vengono aggiunte al modello utilizzando le espressioni DAX. Le colonne calcolate sono compresse durante la fase finale di lavorazione, quando tutte le altre hanno già terminato la loro compressione. Di conseguenza, Vertipaq non le considera nella scelta del miglior ordinamento possibile.[30]

2.3.4 Relazioni in Vertipaq

Le relazioni giocano un ruolo importante nel motore Vertipaq.

Una relazione è una struttura di dati che mappa gli ID di una tabella sui numeri di riga in un'altra. Per migliorare le prestazioni delle query, Vertipaq memorizza le relazioni come coppie (ID, numeri di riga).

Dato l'ID può trovare immediatamente le righe corrispondenti nella tabella, rispetto alla relazione in esame.

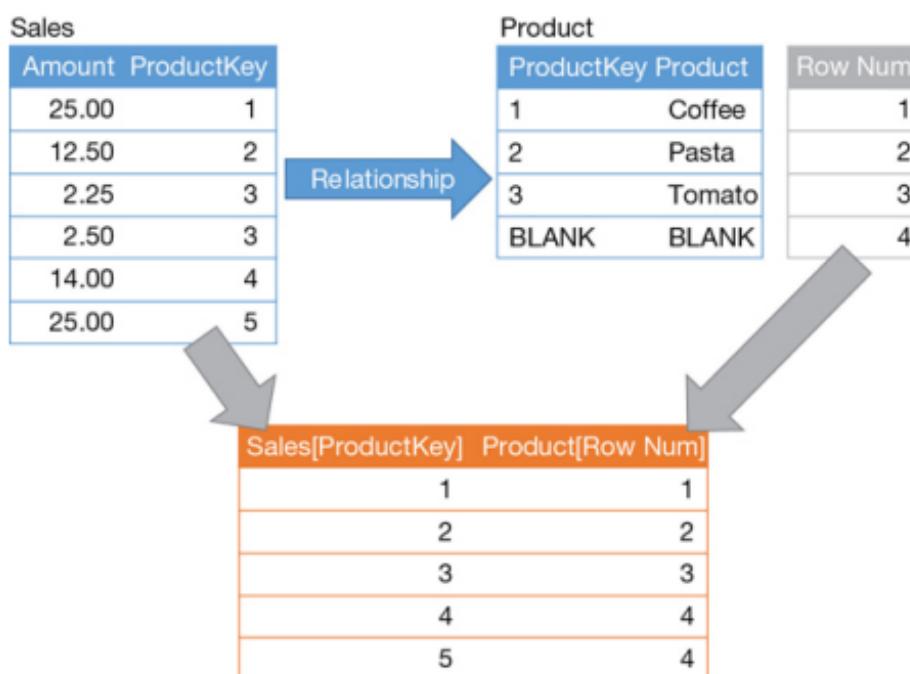


Figura 2.5: Esempio relazione

Le relazioni potrebbero influire sulle prestazioni di una query, nonostante la maggior parte dei calcoli avviene nel motore SE.

Il costo di una relazione dipende dalla cardinalità della colonna che definisce la relazione. Più è bassa, meglio è!

Quando la cardinalità di una relazione è superiore a un milione di valori univoci, l'utente finale può riscontrare prestazioni più lente. Le aggregazioni in Vertipaq possono mitigare l'impatto delle relazioni ad alta cardinalità, preag-

gregando i dati con una granularità diversa.

2.3.5 Ottimizzazioni in Vertipaq:

I concetti di materializzazione e aggregazione dei dati

E' bene porre l'accento su due processi fondamentali implementati in Vertipaq: la materializzazione e l'aggregazione dei dati. La materializzazione è una fase dell'esecuzione della query tipica nei database a colonne.

Il principio di base è che, ogni volta che il motore FE invia una richiesta al motore SE, riceve una tabella non compressa generata dinamicamente: la datacache.

Un grande processo di materializzazione si verifica quando una singola query del motore SE produce una grande cache di dati. Ogni qual volta che il motore SE non è in grado di eseguire tutte le operazioni richieste dalla query DAX, FE eseguirà il lavoro utilizzando una copia dei dati di proprietà del motore SE.

Esistono diversi tipi materializzazione:

- **precoce:** il motore SE produce più datacache, eseguendo operazioni di unione o raggruppamento;
- **tardiva:** il motore SE produce una singola datacache con la cardinalità della query DAX in esame, non avendo la necessità di aggregare i dati.

Prevedere la materializzazione non è facile senza una profonda conoscenza del motore Vertipaq. In query complesse, è quasi impossibile ottenere una materializzazione tardiva ottimale. Pertanto, lo sforzo per ottimizzare una query sta nello spingere la maggior parte del computazionale al motore SE.

Un modello di dati può avere più tabelle correlate ai dati grezzi originali, con lo scopo di offrire modi alternativi al motore di archiviazione per recuperare i

dati più velocemente.

Le tabelle utilizzate a questo scopo sono chiamate aggregazioni. Un'aggregazione non è altro che una versione preraggruppata della tabella originale, con un minor numero di colonne (quindi anche di righe), ottenuta sostituendo i valori con il loro aggregato.[23]

I tipi funzioni di aggregazione disponibili, per righe, sono:

- **conteggio;**
- **minimo;**
- **massimo.**

E' bene specificare che non è possibile applicare una funzione di aggregazione su una colonna calcolata. Una tabella in un modello tabulare può avere più aggregazioni con priorità diverse. Inoltre, aggregazioni e tabelle originali possono essere archiviate con diversi motori.

Le aggregazioni sono potenti, richiedono, però, molta attenzione ai dettagli. Un modo errato di aggregare potrebbe produrre risultati non corretti; deducendo che è uno strumento di ottimizzazione da usare laddove è strettamente necessario, avendone prima constatato un reale beneficio nel contesto di applicazione in esame.

2.4 Azure Analysis Services

2.4.1 Premessa: Piattaforma PaaS

Una piattaforma distribuita come servizio (PaaS, Platform as a Service) è un ambiente di sviluppo e distribuzione completo nel cloud. Le applicazioni ed i servizi sviluppati sono gestiti dall'utente, il cloud gestisce tutto il resto.

Il modello PaaS consente di evitare le spese e la complessità legate all'acquisto e alla gestione di licenze software, middleware, agenti di orchestrazione di contenitori come Kubernetes o strumenti di sviluppo e altre risorse.[24]

Questa architettura è progettata per supportare il ciclo di vita completo delle applicazioni Web: creazione, test, distribuzione, gestione e aggiornamento, offrendo una riduzione del tempo per la scrittura di codice grazie a componenti pre-codificati integrate nella piattaforma.

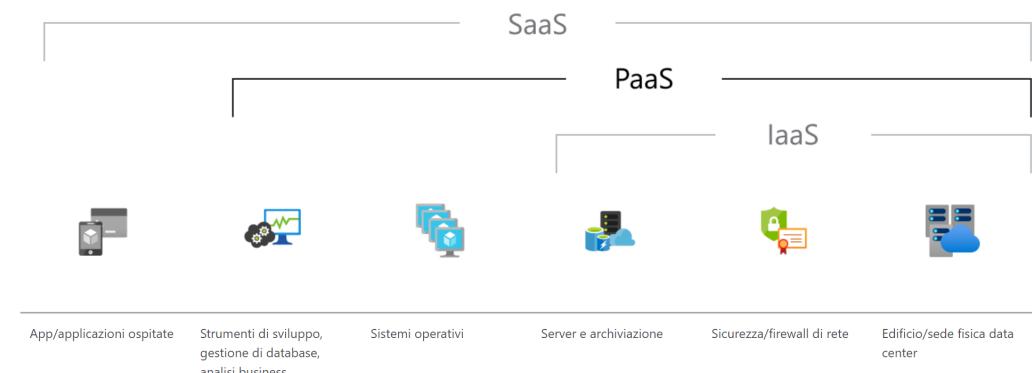


Figura 2.6: Architettura PaaS

Alla luce di tutto ciò, Azure Analysis Service è un'implementazione di questa architettura fornita nel pacchetto Microsoft SQL Server Data Tools.

2.4.2 Modello di Sicurezza in Azure Analysis Service

La sicurezza è basata sui ruoli, associati a singoli utenti o ad interi gruppi.

Il meccanismo di sicurezza principale è un ruolo, a cui possono essere assegnate autorizzazioni a livello di modello, filtri di riga e controlli.

Esistono diversi ruoli che sono lettura, lettura ed elaborazione, solo elaborazione, amministratore.

Tutto questo ragionamento si riflette sui filtri, determinati in base al ruolo assunto. I filtri di riga offrono la possibilità di filtrare l'intero modello, quando gli utenti del ruolo visualizzano i dati. Gli altri filtri sono costruiti per consentire agli utenti la possibilità di vedere un sottoinsieme di un modello più grande, puramente per evitare che un numero eccessivo di oggetti venga visualizzato nel visualizzatore del modello.

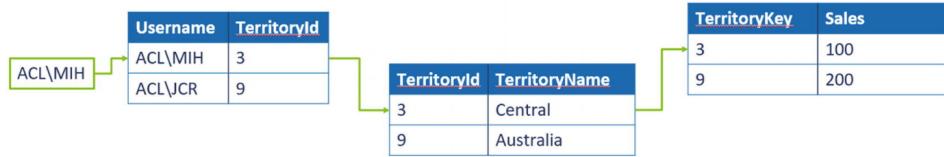


Figura 2.7: Sicurezza Dinamica

2.4.3 Elaborazione di un modello

Azure Analysis Service offre la possibilità di ridimensionare il modello per soddisfare le esigenze di elaborazione e di query, implementando il motore Vertipaq analizzato precedentemente.

Una volta sviluppato, conterrà non solo dati ma anche oggetti di analisi come indicatori, colonne calcolate e gerarchie. Ovviamente, i dati contenuti nel modello non rimarranno fermi. Devono essere aggiornati e ricalcolati regolarmente per garantire che siano consistenti.

I modelli tabulari di Azure Analysis Service vengono aggiornati facilmente uti-

lizzando protocolli comuni come REST o cmdlet di PowerShell.

Esistono vari livelli di elaborazione:

- **predefinito:** elabora tutte le tabelle non ancora processate e calcola colonne e gerarchie. A livello di partizione, vengono eseguiti gli stessi passaggi esclusivamente per partizionare gli oggetti.
- **completo:** esegue un processamento completo di tutti gli oggetti nel modello, partizione o tabella a seconda dell'opzione di elaborazione, oltre al calcolo di colonne e gerarchie.
- **relativo ai dati:** elabora semplicemente i dati nel modello, nella partizione o nella tabella, non ricalcolando, tuttavia, le colonne o le gerarchie.
- **di eliminazione:** cancellando i dati da un modello, partizione o tabella.

2.5 Power BI

Power BI è il prodotto di punta per la visualizzazione dei dati di casa Microsoft. Il sistema implementa lo stesso motore analitico utilizzato per l'Azure Analysis Service, ottimizzando le query analitiche sui dati tabulari tramite l'elaborazione in memoria che, abbinata a un ricco set di funzionalità di visualizzazione, consente agli sviluppatori di ottenere facilmente varie tipologie di grafici (report).

In particolar modo, Power BI offre una piattaforma di visualizzazione di alto livello, garantendo la capacità di stratificare i dati aziendali tramite apposite funzionalità e garantendo un alto livello di efficienza nell'individuazione delle

falle e nell'ottimizzazione dei processi. Vengono generati report regolari, automatici o manuali, garantendo a ciascun utente di generare i più adatti al proprio utilizzo.

Troviamo diverse componenti chiave [30]:

- **Power BI desktop:** E' lo strumento di sviluppo principale per i file, gratuito per tutti gli sviluppatori di report.
- **Power BI report builder:** E' il generatore utilizzato per creare report impaginati anziché dashboard.
- **Power BI service:** E' il portale web in cui vengono pubblicati dashboard e report, accessibile agli utenti con licenze Power BI Pro ed esteso a dispositivi mobili come telefoni e tablet.

I dati per Power BI possono essere derivati da un'ampia gamma di fonti e combinati insieme per formare set di dati consolidati.

Ciò potrebbe significare combinare dati pubblici con dati interni di un data warehouse o analizzare più fogli Excel insieme a file in un data lake. Il tutto avviene mentre si è connessi ad Azure Analysis Services, garantendo al front-end di Power BI di inviare le query al motore di Analysis Service, senza lunghe operazioni di importazione.

2.5.1 Architettura

L’architettura sotto rappresentata, utilizza una metodologia di processing dei dati ETL, come specificato nel capitolo iniziale.

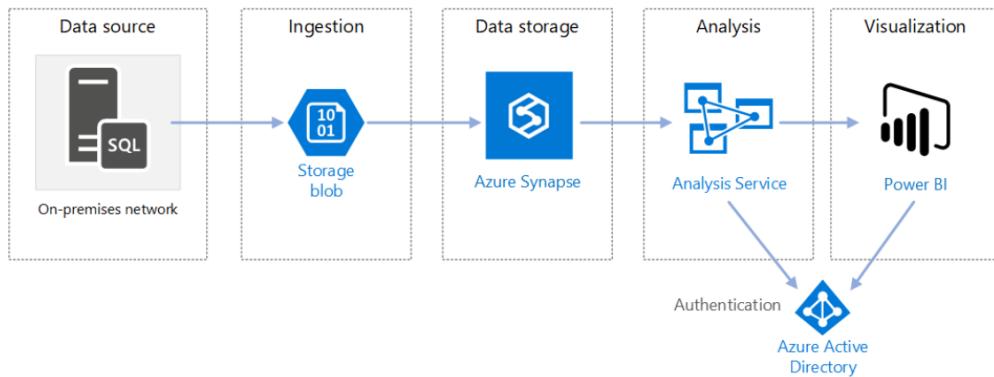


Figura 2.8: Sicurezza Dinamica

Entrando più nel dettaglio nella descrizione delle componenti, abbiamo diversi moduli:

- **Database SQL Server**: per simulare l’ambiente locale, gli script di distribuzione per questa architettura eseguono il provisioning di una macchina virtuale in Azure con SQL Server installato.
- **Inserimento e archiviazione dei dati**: l’archiviazione BLOB viene usata come area di gestione temporanea per copiare i dati, prima di caricarli in sinapsi di Azure.
- **Sinapsi di Azure**: Azure sinapsi è un sistema distribuito progettato per eseguire analisi su dati di grandi dimensioni. Supporta l’elaborazione parallela su larga scala (MPP), che può essere usata per l’esecuzione di analisi ad alte prestazioni.

- **Azure Analysis Service:** il sistema implementa l’Azure Analysis Service, dettagliato nella sezione precedente, leggendo i dati dal data warehouse per elaborare il modello semantico e gestire in modo efficiente le query. I modelli tabulari usano costrutti di modellazione relazionale (tabelle e colonne), mentre i modelli multidimensionali usano costrutti di modellazione OLAP (cubi, dimensioni e misure).

Power BI supporta due opzioni per la connessione ad Azure Analysis Service:

- I dati sono importati nel modello Power BI.
- I dati vengono estratti direttamente da Azure Analysis Service tramite una connessione dinamica poiché non richiede la copia dei dati nel modello di Power BI.

Si noti che, per sfruttare al meglio questa architettura, si deve evitare di eseguire query dashboard BI direttamente sul data warehouse.

- **Autenticazione:** Basata sui ruoli, come descritta nella sezione precedente.

2.5.2 KPIs

Una feature molto importante, all’interno di Power BI, è rappresentata dai Key Performance Indicators (KPIs), essenziali nei reports.

I KPI valutano le prestazioni di un’organizzazione in diverse aree, analizzando valori di misura quantificabili e svolgendo un ruolo essenziale, tramite un impatto visivo. Un KPI ha sempre un valore di base o una misura che viene valutata rispetto a un valore target, confrontando le prestazioni effettive con l’obiettivo prefissato.

Abbiamo KPIs di alto livello che mostrano le prestazioni complessive di un'organizzazione fornendo una visione generale e KPIs di basso livello, che si concentrano sui singoli aspetti come dipendenti, vendite, marketing, produzione, ect.

Ogni KPI in Power BI può essere suddiviso in tre parti:

- **Valore di base:** un valore di base è una media della somma delle vendite, del profitto lordo, ect.
- **Valore target:** un valore target è una misura che è un valore assoluto, impostato come un obiettivo rispetto al quale viene valutato un valore target.
- **Soglie di stato:** le soglie di stato forniscono un intervallo per valutare il valore di base e i valori di destinazione. L'oggetto visivo presenta anche un intervallo massimo e minimo tra i quali rientrano questi valori.

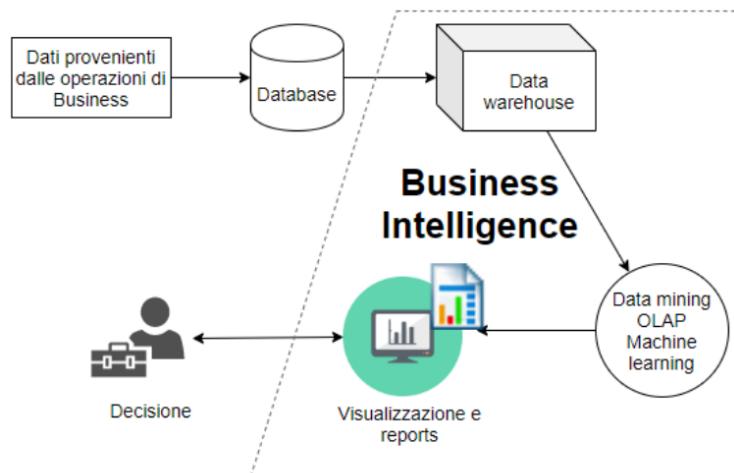


Figura 2.9: Visualizzazione reports e fasi antecedenti

Si può concludere che lavorare con i KPI facilita la visualizzazione di informazioni di per sé complesse, ricavate da tutte le fasi antecedenti alla reportistica finale.

Capitolo 3

L’Intelligenza Artificiale nel mondo del Business

3.1 Visualizzazione dei dati: perchè è importante?

Tecnicamente parlando, la Business Intelligence coinvolge un’ampia categoria di applicazioni e tecnologie per la raccolta, memorizzazione, analisi ed accesso ai dati, in modo tale da aiutare i clienti a prendere le decisioni aziendali migliori.

Le funzioni, ramificate in questa area, includono reporting, elaborazione analitica online, analisi, data mining, prestazioni aziendali, gestione, benchmarking, text mining e predizioni. Utilizzando strumenti di illustrazione e progettazione grafica, i dati possono essere visualizzati. Lo scopo principale consiste nel comunicare le informazioni in modo chiaro ed efficace. [22]. L’idea è quella di creare dati sia estetici che funzionali, visualizzazioni al fine di fornire approfondimenti e modi intuitivi di percepire dati complessi.

Ricapitolando, attuando questo modus operandi, si aiutano le persone a capire

l'importanza dei dati, riassumendo e presentandone un'enorme quantitativo, in un formato semplice e di facile comprensione.

La visualizzazione dei dati, aiuta l'azienda a raggiungere i propri obiettivi[21]:

- Convertire dati in grafici interattivi permette un'interpretazione dinamica degli scenari tipici aziendali.
- Prendere decisioni appropriate, approfondendo i dati e trovando modelli, tendenze e correlazioni, utili a determinare dove migliorare i propri processi operativi.
- Fornire un quadro più completo dei dati in analisi.
- Prendere decisioni migliori, rapide e informate con i dati visualizzati.

3.2 Image Recognition: la nuova frontiera dell'IA

La consapevolezza del valore dei dati sta portando a una trasformazione dei processi aziendali. L'impiego dell'AI trova così sempre più spazio nei progetti di business, con l'image recognition e l'image detection tra le declinazioni a maggior tasso di crescita[6]. La trasformazione digitale è il fenomeno che sta interessando a ritmi sostenuti il mondo del business, potendo intervenire in diversi ambiti in maniera più o meno significativa. Un fenomeno che non lascia inesplorato nessun aspetto e nessuna fonte: dalle immagini, con l'image recognition e l'image detection, la voce, con il voice recognition, oltre ovviamente allo sfruttamento dei vari dati strutturati e non che passano spesso inosservati nella rete, dai social network alle informazioni residenti nelle aziende stesse.

Tutti i dati sono quindi “appetibili” per poterci ricavare delle informazioni. Devono essere tradotti in forma digitale, tali da poter essere processati, analizzati e interpretati dai nuovi sistemi basati sulle nuove tecnologie, in grado di mettere in moto algoritmi per la loro interpretazione[6]. Le tecnologie di data science, impongono la necessità di passare da grafici relativamente semplici a mappe relazionali multidimensionali, aumentando così il proprio peso.[21]. L’intelligenza artificiale riesce attraverso classificatori di vario tipo, soprattutto attraverso reti neurali evolute, a riconoscere le immagini. L’evoluzione di questi reti, infatti, sta tendendo ad un livello tale da riuscire in qualche modo a simulare il comportamento cognitivo del cervello umano nei confronti della vista, rendendo la image recognition e la image detection dei processi molto simili a quelli umani. Esperimenti sul cervello hanno, infatti, osservato che la corteccia visiva primaria è composta da un insieme di strutture neuronali semplici e che è proprio utilizzando tali strutture che si attiva la vista[6]. Sulla base di questa similitudine, si sviluppano gli algoritmi per la lettura e interpretazione delle immagini anche da parte delle macchine. Grazie alle reti neurali, si riescono a riconoscere forme, colori, fino addirittura a seguire oggetti in movimento. Si apre così la strada a una grande varietà di applicazioni, con un mercato promettente[6].

3.3 Che cos’è un’immagine digitale?

Un’immagine è una rappresentazione bidimensionale dello spettro di luce visibile. Un’immagine digitalizzata è, invece, una griglia di pixel, piccoli quadratini con associato un colore.

Come vengono rappresentati i colori di ogni pixel?

Nel caso più semplice, le immagini in bianco e nero, ogni pixel è un numero, 0 o 1, a seconda del colore. L’immagine è una matrice di zeri e uni. Ammettendo numeri tra 0 e 255, possiamo costruire un’immagine in scala di grigi: il nero è 0, il bianco 255. Più alto sarà il numero inserito, più il pixel sarà ’chiaro’.

Viceversa, più basso sarà il numero inserito, più il pixel sarà 'scuro'.

Per introdurre i colori esistono diversi formati: uno semplice è quello RGB (Red Green Blue). Non avremo una matrice di numeri ma una matrice di tuple, dove il primo numero indica l'intensità del rosso, il secondo l'intensità del verde ed il terzo l'intensità del blu. Anche qui, ogni valore va da 0 a 255. La scelta di 255 come massimo non è casuale: contando da 0 abbiamo 256 gradazioni possibili per ogni colore. RGB non è l'unica codifica esistente, ci sono diverse alternative. A volte nelle immagini è presente un quarto valore, detto alpha, che indica la trasparenza del singolo pixel. Altri formati si basano su parametri come brillantezza del colore o cose simili. Il concetto importante è, però, che ci si sta occupando di una matrice di tuple righe per colonne, o al massimo di una matrice di numeri. Questi numeri, pur essendo punti molto piccoli, ci danno la sensazione di colori continui appena vengono visualizzati.

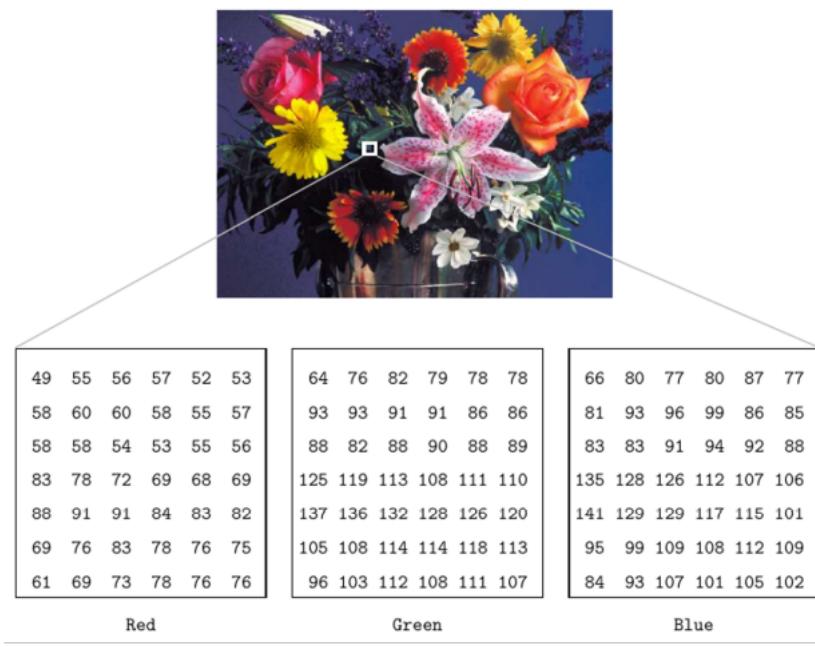


Figura 3.1: Immagine RGB

3.4 Ingegnerizzazione delle Immagini

Una disciplina relativamente nuova e attualmente in crescita, consiste nello studio di vari problemi e nello sviluppo di una tecnologia efficiente nel campo dell'immagine applicata . Questo quadro generale, che combina i principi delle scienze di base come la matematica e l'ottica con le tecnologie o le applicazioni dell'immagine, offre un contenuto molto ricco.

Otteniamo una metodologia che combina analisi di immagini più comprensione delle stesse, includendo applicazioni ingegneristiche [35].

3.4.1 Image Preprocessing

Elaborare, analizzare e comprendere formalmente le immagini, prima che vengano eseguite. La preelaborazione tipica, include la cancellazione del rumore, l'estrazione delle caratteristiche, la registrazione dell'immagine, la correzione e la normalizzazione del target di interesse etc.

Lo scopo della preelaborazione è migliorare la qualità dell'immagine per renderla più "appetibile" ad un calcolatore. A seconda dello spazio in cui viene eseguita o localizzata, è possibile distinguere operazioni spaziali o trasformazioni. Le prime citate, possono ulteriormente essere suddivise in operazioni globali (operazioni a immagine intera), operazioni locali (operazioni adiacenti) e operazioni multiple su immagine. Le trasformazioni, possono anch'esse essere suddivise in operazioni nel dominio della frequenza (tramite trasformata di Fourier), operazioni nel dominio wavelet e così via [35].

3.4.2 Image Analysis

Processo per ricavare informazioni oggettive, stabilendo una descrizione dell'immagine e dell'obiettivo da perseguire. Il modello matematico è combinato con la tecnologia di elaborazione delle immagini, per analizzare le caratteristiche sottostanti e la struttura di livello superiore, in modo da estrarre in-

formazioni con una certa percezione visiva e semantica. Se l'elaborazione è un processo da immagine a immagine, l'analisi è un processo da immagine a dati. Qui, i dati possono essere il risultato di una misurazione caratteristica dell'oggetto, o basati su una certa rappresentazione simbolica della misurazione, che descrive le caratteristiche e le proprietà dell'oggetto nell'immagine. Il risultato di un'operazione di analisi generale dell'immagine, è una descrizione quantitativa del contenuto della stessa[35].

3.4.3 Image Understanding

Il focus è, sulla base dell'analisi dell'immagine, utile a studiare ulteriormente l'interrelazione tra questa ed i vari target. Attraverso la comprensione del contenuto, si cerca di ottenere una spiegazione della scena oggettiva, originale e che guida nella pianificando dell'azione.

La comprensione è, in una certa misura, centrata sul mondo oggettivo, includendo cose che non sono direttamente osservabili, che piuttosto devono essere colte tramite conoscenza e l'esperienza[35].

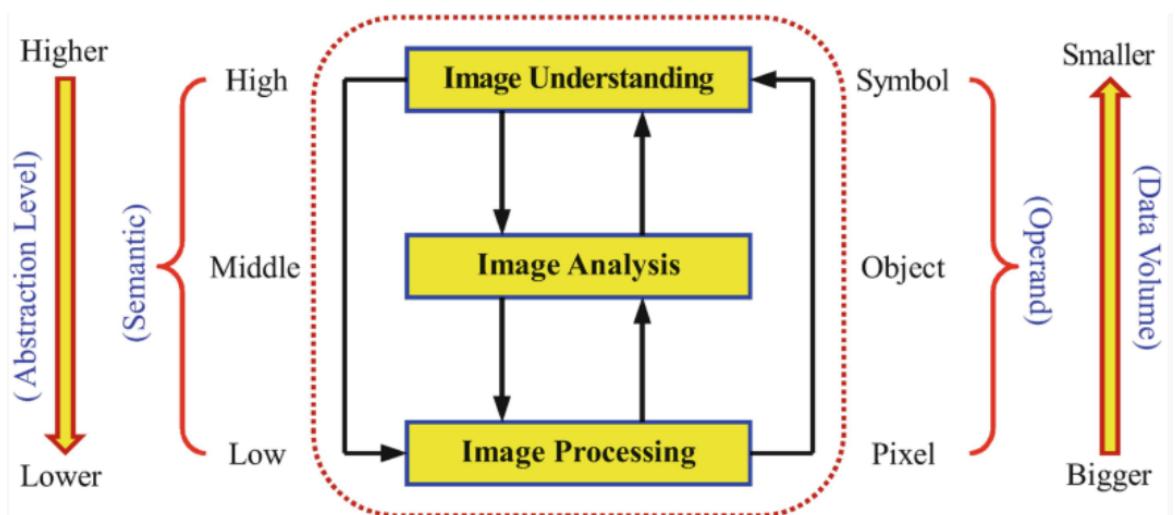


Figura 3.2: Fasi di Ingegnerizzazione dell'immagine

Capitolo 4

Tecniche di classificazione delle immagini

Il riconoscimento e la classificazione delle immagini, è ciò che consente molti dei risultati più impressionanti dell'intelligenza artificiale. Ma come imparano i computer a rilevarle e classificarle?

Vediamo le principali tipologie di classificatori e reti utilizzati.

4.1 Algoritmi di Machine Learning

Esistono molti tipologie di algoritmi di machine learning, permettendo una divisione in diverse categorie, in base al loro utilizzo.

- **Apprendimento supervisionato**

Un algoritmo è supervisionato quando, sulla base delle info ricavate da un dataset etichettato ricevuto come esempio, apprenderà come classificarne uno completamente nuovo, non etichettato. Quando si parla di algoritmi per il machine learning, dove l'etichetta è di tipo testuale, si parla di classificazione. Nel caso in cui fosse numerica, di regressione.

- **Apprendimento non supervisionato**

Le tecniche di apprendimento non supervisionato, invece, si basano fondamentalmente sul clustering o sulle regole associative. Nel primo caso, si inizia da un dataset non etichettato nel quale, a partire dalle caratteristiche di un gruppo di elementi scelti, si aggregano i restanti in un certo numero di cluster prefissato. Il tutto è regolato da varie logiche come similarità e distanza.

Esistono, inoltre, algoritmi distinti sulla base delle logiche sottostanti: modelli logici come alberi decisionali, modelli probabilistici come Bayes o geometrici, come la regressione.

Per cominciare a utilizzare gli algoritmi per il machine learning, è necessario inquadrare il nostro progetto nella categoria corretta: classificazione, clustering, previsioni numeriche o riconoscimento dei pattern[5].

4.1.1 AdaBoost

Il Boosting è un metodo di ensemble per migliorare le previsioni del modello di un determinato algoritmo di apprendimento, tramite la formazione sequenziale di weak learner, ciscuno con lo scopo di cercare di correggere il suo predecessore.

Un weak leaner, è un classificatore che funziona male ma meglio rispetto ad un'ipotesi casuale.

In prima istanza, viene creato un modello dai dati di addestramento, aggiungendone altri fino a quando il set non viene previsto perfettamente o viene raggiunto il numero massimo di modelli stabilito.

Adaptive Boosting rappresenta una tecnica di boosting popolare che sfrutta questo approccio avvalendosi di tanti alberi decisionali con un determinato livello di profondità, definiti decision stumps, tanti quante sono le caratteristiche del modello[10].

Ad ogni iterazione, un nuovo classificatore debole viene introdotto in sequenza, mirando a compensare le carenze dei modelli precedenti con lo scopo finale di creare un classificatore forte. E' bene specificare che qualsiasi algoritmo di apprendimento automatico, che accetta pesi sul set di train, potrebbe essere utilizzato come classificatore di base.

- **1. Calcolo del peso di ogni data point**

Dato un set di n punti, con $X_i \in R^d, y \in \{-1, 1\}$ con classe 1 positiva e -1 negativa, inizializzo il peso per ciascun punto con $w(x_i, y_i) = 1/n, i = 1, \dots, n;$

- **2. Scelta del weak learner**

Un Decision Stump è un albero di classificatori dove ogni nodo interno è associato ad una particolare “domanda” su una caratteristica (feature).

Da questo nodo dipartono tanti archi quanti sono i possibili valori che la caratteristica può assumere, fino a raggiungere le foglie che indicano la categoria associata alla decisione. Ipotizzo un decision stump con $\text{depth}=1$, formandone uno per ogni caratteristica e scegliendo il weak learner che minimizza l'entropia.

- **3. Calcolo dell'Amount of Say (performance) del weak learner scelto in precedenza**

Per capire come trovare questo valore si usa il Total Error (TE), dato dal numero di previsioni errate rispetto al totale dei data points utilizzati.

- **4. Calcolo del peso delle nuove istanze**

Per le classificazioni errate è $WE_{new} = \frac{w_{old}}{2*TE}$, mentre per quelle corrette si ha $WC_{new} = \frac{w_{old}}{2*(1-TE)}$.

- **5. Creazione di un nuovo dataset tramite i nuovi pesi**

Si divide il dataset in una distribuzione di valori compresa tra 0 e 1, utilizzando i pesi appena creati e costruendo un nuovo dataset della stessa grandezza del precedente.

- **6. Ripetizione dei passi da 2 a 5**

Si ripetono i passi da 2 a 5 n volte, fino a quando i dati di addestramento non ottengono nessun miglioramento, oppure fino al raggiungimento del numero massimo di estimatori specificato. Si noti che, per prima cosa, il peso dei nuovi campioni viene aggiornato ad ogni iterazione. Inoltre, l'errore totale (TE) cambierà a seconda delle nuove previsioni fatte dal decision stump. Il peso dei nuovi campioni per l'iterazione i-esima, influenzera il valore del peso all'iterazione successiva (i-esima +1), poichè l'algoritmo tiene conto dei pesi ottenuti dalle iterazioni precedenti. Col valore di quest'ultimi pesi, si riforma un nuovo dataset e si riparte con la creazione di un nuovo decision stump, fino a quando la condizione di stop non viene raggiunta[10].

- **7. Costruzione del dataset finale**

Una volta completate tutte le iterazioni, l'insieme dei weak learner viene combinato, insieme ai loro pesi, per formare uno strong learner.

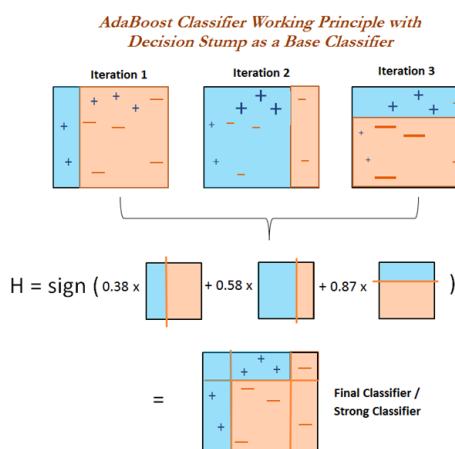


Figura 4.1: Classificatore AdaBoost

4.1.2 Support Vector Machine

Il Support Vector Machine ha l'obiettivo di identificare l'iperpiano che meglio divide i vettori di supporto in classi, punti dati più vicini all'iperpiano che, se rimossi o modificati, alterano la posizione dell'iperpiano divisorio[14]. Il metodo, va alla ricerca di un iperpiano linearmente separabile o di un limite di decisione. Entrambi i modi separano i valori di una classe dall'altra scegliendo, se ne esiste più di uno, quello che ha margine più alto in termini di accuratezza con i vettori di supporto. Se tale iperpiano non esiste, SVM utilizza una mappatura non lineare per trasformare i dati di allenamento in una dimensione superiore, permettendo sempre una suddivisione. Questo tipo di classificatore è spesso utilizzato per testo, immagini, volti etc.

SVM per modello lineare

Il problema consiste nel trovare quale tra le infinite rette possibili risulti ottimale, generando il minimo errore di classificazione su una nuova osservazione. Infatti, più lontano dall'iperpiano si trovano i punti dati, maggiore è la possibilità che siano classificati correttamente.

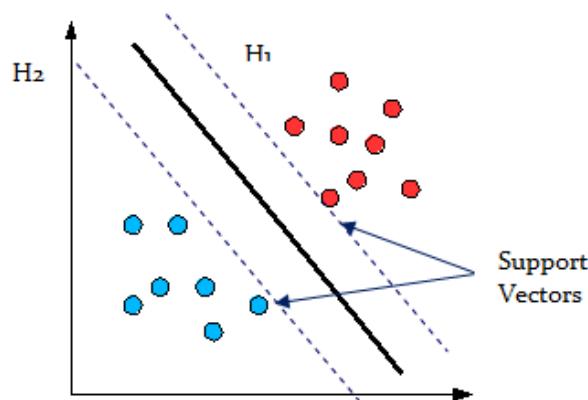


Figura 4.2: Iperpiano lineare

L'iperpiano ottimale, è definito come $\vec{w}\vec{x} + w_0 = 0$, dove w è il vettore dei pesi, x il vettore delle caratteristiche di input e w_0 è il bias. In altre parole, in n dimensioni un iperpiano di separazione è una combinazione lineare di tutte le dimensioni poste uguali a 0. Volendo includere in queste condizioni anche i limiti di margine delle classi, è possibile regolare i coefficienti o i pesi w_1 e w_2 nella seguente forma:

$$w_0 + w_1x_1 + w_2x_2 \geq 1 \text{ se } y=1 \text{ (etichetta positiva)}$$

$$w_0 + w_1x_1 + w_2x_2 \leq -1 \text{ se } y=0 \text{ (etichetta negativa)}$$

I due risultati rappresentano i confini del margine (sono i valori tratteggiati nell'immagine seguente e sono anch'essi due iperpiani). I dati di addestramento, che ricadono esattamente sui confini del margine, sono chiamati vettori di supporto. Minimizzando il vettore peso w avremo margine massimo, determinando l'iperpiano ottimale [14].

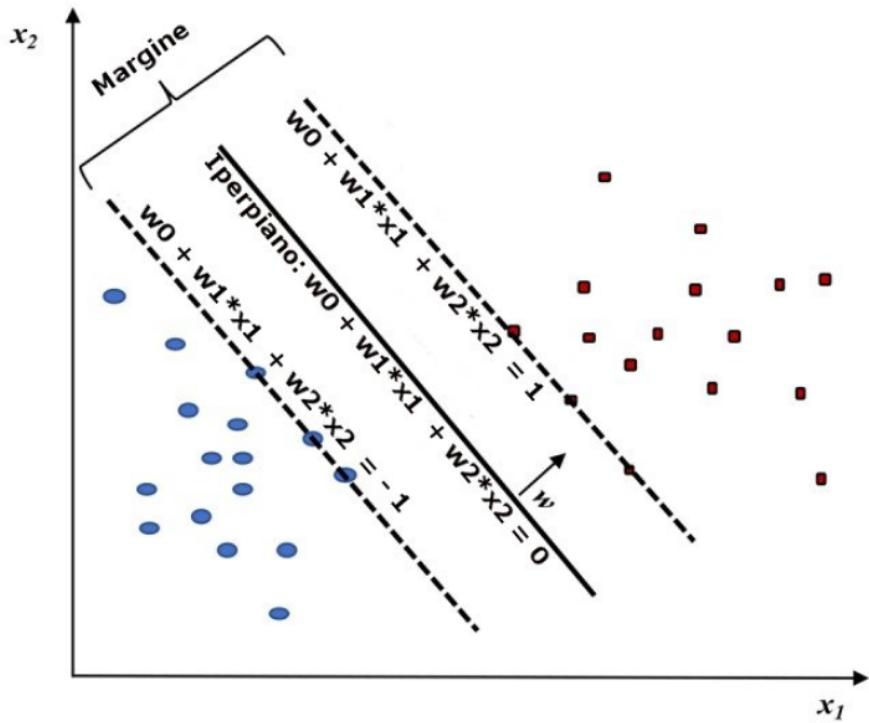


Figura 4.3: Risultato ottimale

Dataset non lineare e funzione kernel

Il metodo del kernel, consente di lavorare con modelli non lineari e di dimensioni superiori.

In generale, ricordiamo che dati $(x^i, y^i), i = 1 \dots, l$, con $x^i \in X, y^i \in -1, 1$, il kernel può essere definito come una funzione simmetrica $K : X \times X \rightarrow R$, tale che esistono uno spazio lineare H feature space in cui è definito il prodotto scalare $\langle ., . \rangle$ e una funzione $\phi : X \rightarrow H$ per cui $k(x, y) = \phi(x), \phi(y)$.

Una scelta appropriata di questa funzione, impatta notevolmente sulle prestazioni del classificatore. Le funzioni kernel più diffuse sono:

- **Kernel lineare:**

$K(x^i, x^j) = x^{i^T} x^j$, con $H = R^n$. Il prodotto scalare è quello usuale tra due vettori, ϕ è definito come $\phi(x) = x$, con $\phi : R^n \rightarrow R^n$ adatto alla classificazione testuale.

- **Kernel polinomiale:**

$K(x, y) = (ax^T y + b)^p$, con $a, b \in R$ e $p \geq 1$ che rappresenta i gradi di libertà, da controllare opportunamente per evitare overfitting.

- **Kernel RBF :**

$K(x^i, x^j) = e^{-\gamma \|x^i - x^j\|^2}$, chiamato anche kernel gaussiano. Il kernel RBF, contiene un parametro γ : un piccolo valore di γ farà sì che il modello si comporti come un SVM lineare, mentre un grande valore di γ renderà il modello fortemente influenzato dagli esempi dei vettori di supporto[14].

Vantaggi e Svantaggi

SVM è uno strumento efficace negli spazi ad alta dimensione e nel consumo di memoria. Infatti, un sottoinsieme dei punti di addestramento viene utilizzato nel processo decisionale effettivo di assegnazione di nuovi membri, considerando solo quest'ultimi ai fini decisionali. Inoltre è versatile, grazie alla separazione di classi è altamente non lineari, consentendo una sostanziale flessibilità per i limiti decisionali, con conseguente maggiore performance di classificazione.

Appartenendo alle tecniche non parametriche, SVM manca di trasparenza nei risultati, comunque attenuata dalla visualizzazione grafica. Non esiste, inoltre, un'interpretazione probabilistica diretta per l'appartenenza al gruppo, con conseguente verifica dei risultati prodotti.

4.1.3 K-Nearest Neighbors

Uno degli algoritmi più conosciuti nell'ambito del machine learning è il K-Nearest Neighbors (KNN) che, oltre alla sua semplicità, produce buoni risultati in vari domini applicativi[12].

La metodologia utilizza un algoritmo di apprendimento supervisionato, il cui scopo è quello di predire una nuova istanza conoscendo i data points (centroidi), separati in diverse classi. Il suo funzionamento si basa sulla somiglianza delle caratteristiche: più un'istanza è vicina a un data point, più KNN li considererà simili. Solitamente, la distanza euclidea è utilizzata come metrica. Minore sarà la distanza e maggiore sarà la somiglianza tra il punto e l'istanza da prevedere, assegnandolo ad un gruppo relativo a questi.

La scelta di k è cruciale, in quanto l'algoritmo valuta le k minime distanze.

Essendo KNN non parametrico, non si fa alcuna ipotesi sulla distribuzione dei dati analizzati. Ricapitolando, la struttura del modello è determinata dai dati ed è piuttosto utile. Infatti, la maggior parte dei dati non obbedisce ai tipici assunti teorici fatti. Di conseguenza potrebbe, e probabilmente dovrebbe,

essere una delle prime scelte per uno studio di classificazione, in presenza di poca o nessuna conoscenza a priori sulla distribuzione dei dati[12].

Se si conoscono i gruppi che si vogliono visualizzare come output, si può partire impostando k pari a tale numero. Altrimenti, il metodo più tradizionale e diretto è iniziare con un k casuale, creando k centroidi casuali ed eseguendo l'algoritmo.

Algoritmo KNN

- 1. Inizializzazione

Si sceglie un valore k con cui prevedere il nuovo data point.

- 2. Preprocessing

Nel caso di grandezze non comparabili, si utilizzano tecniche per rendere le misure confrontabili, come la normalizzazione, altrimenti si passa al punto 3.

- 3. Calcolo Distanza

Si calcola la distanza (ad esempio quella euclidea) tra la nuova istanza e i vari data points.

- 4. Ordinamento

Si ordinano le distanze calcolate dalla più piccola alla più grande.

- 5. Selezione

Si scelgono le prime K : nel caso si stia svolgendo un problema di regressione, si può restituire la media delle etichette K . Se si sta svolgendo un problema di classificazione, si sceglierà la classe che include più valori k trovati precedentemente.

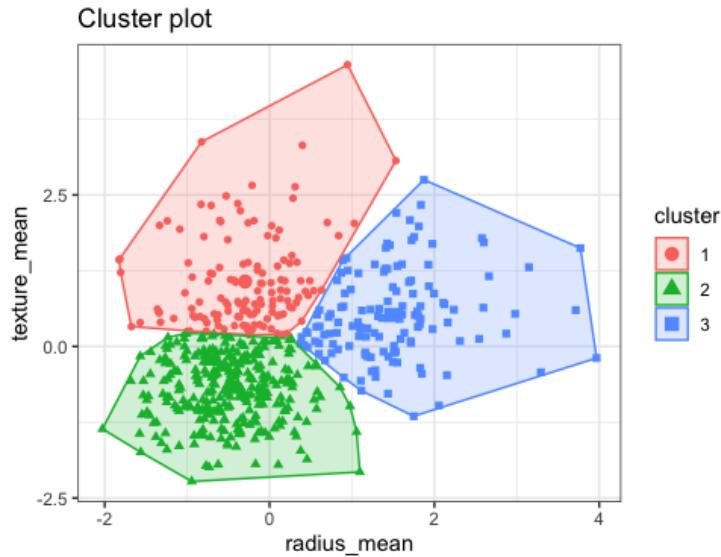


Figura 4.4: Esempio Algoritmo KMeans

Scelta di K e Cross Validation

Un aspetto cruciale, per la buona riuscita della previsione di k, è l'utilizzo del metodo più appropriato. Ad esempio, troviamo la convalida incrociata o cross validation, metodo statistico utilizzato per stimare l'abilità dei modelli di apprendimento automatico. La convalida viene definita dal processo su cui il modello addestrato viene valutato, tramite un set di dati di test, parte separata dello stesso set di dati da cui deriva il set di training. Lo scopo principale è testare la capacità di generalizzazione di un modello addestrato[8].

In definitiva, la cross validation mira a trovare il modello ottimale con le migliori prestazioni.

Esistono diverse tipologie di Cross Validation:

- **Holdout:** il tipo più semplice di convalida dei dati, con un'unica suddivisione.

- **LOOCV**: il numero delle suddivisioni è pari al numero delle osservazioni nel dataset.
- **K-fold CV**: la convalida più diffusa che permette di definire k suddivisioni del dataset.
- **Stratified Cross Validation**: ad ogni piega o suddivisione, viene mantenuta costante la distribuzione dei campioni tra le classi.
- **ShuffleSplit**: un metodo ibrido tra il metodo holdout e la convalida k-fold.

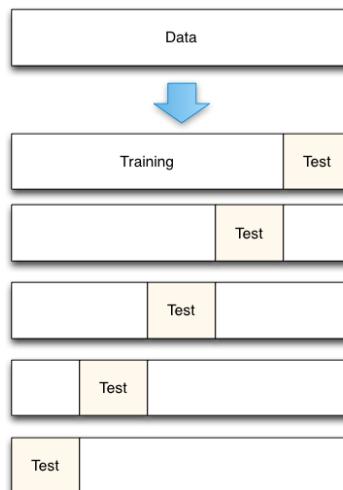


Figura 4.5: K-fold

L’idea generale di questo metodo è quella di dividere il campione di dati in un numero k definito di sottocampioni o segmenti ad estrazione casuale, disgiunti, che serviranno come fase di allenamento (training). Si applica poi il modello KNN per fare previsioni sul segmento k-esimo (ad esempio, utilizzando i segmenti k-1 come esempi) e si valuta l’errore. Ogni volta uno dei sottoinsiemi k

viene usato come set di test, mentre gli altri sottoinsiemi $k-1$ sono messi insieme per formare un set di allenamento. Come si può vedere, ogni data point può trovarsi in un set di test esattamente una volta e può trovarsi in un set di allenamento $k-1$ volte[12]. Alla fine della procedura, gli errori calcolati vengono mediati per fornire una misura della stabilità del modello (ossia quanto bene il modello prevede nuove istanze). I passaggi precedenti, vengono ripetuti per vari k e il valore che raggiunge l'errore più basso (o la massima precisione di classificazione) viene così selezionato come valore ottimale per k (ottimale in un senso di convalida incrociata). Si noti come la convalida incrociata sia dispendiosa dal punto di vista computazionale e si dovrebbe essere pronti a far funzionare l'algoritmo per un po' di tempo, specialmente quando la dimensione del campione di esempi è ampia[12]. In alternativa, si può specificare un k ragionevole, qualora si conosca il problema in esame in maniera dettagliata.

4.1.4 Random Forest

Il Random Forest è un algoritmo di apprendimento supervisionato che si avvale del bagging, come metodo di ensamble, e dell'albero decisionale, come modello individuale. Un metodo di ensemble è una tecnica che combina i responsi di più algoritmi di apprendimento automatico, utili a fornire previsioni più accurate rispetto a qualsiasi singolo modello. Il bagging è l'applicazione della procedura bootstrap a un algoritmo di machine learning, con alta varianza. In genere, si utilizza l'albero decisionale, sensibile ai dati forniti in input[13]. Durante il processo di training, ogni albero in una "foresta casuale" impara da un campione, casuale anch'esso, di punti dati. Tramite la tecnicna di bootstrap, i sample verranno utilizzati più volte in un singolo albero.

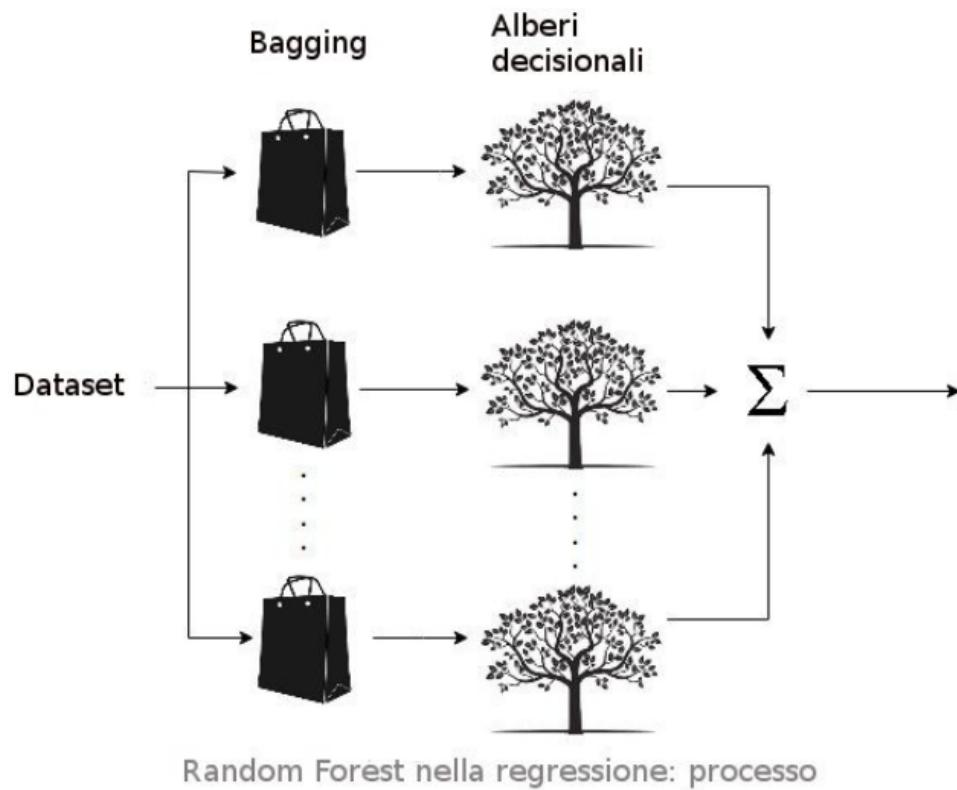


Figura 4.6: Algoritmo Random Forest

L’idea è che si dovrebbe addestrare ciascun albero su campioni diversi, sebbene ogni albero possa presentare una varianza elevata, rispetto a una particolare serie di dati di addestramento. Nel complesso, l’intera foresta avrà una varianza inferiore, ma non a costo di aumentare la distorsione. Al momento del test, le previsioni vengono effettuate calcolando la media di quelle di ciascun albero decisionale[13].

4.1.5 Reti Neurali

Le reti neurali sono modelli di calcolo matematico-informatico. Prendono ispirazione dalle reti neurali biologiche, costituite da interconnessioni, derivanti da neuroni artificiali e processi di calcolo basati sul modello di elaborazione a parallelismo distribuito delle informazioni[4]. Le reti neurali artificiali, sono strutture non-lineari di dati statistici, organizzate come strumenti di modellazione, riceventi segnali esterni su uno strato di nodi, ognuno dei quali è collegato a svariati altri interni della rete. La struttura è organizzata tipicamente su più livelli, in modo che ogni singolo nodo possa elaborare i segnali ricevuti. Ogni livello, trasmette al successivo il risultato delle sue elaborazioni.

Un architettura generale è composta da tre strati:

- **Strato di Input**

Riceve ed elabora i segnali in input, adattandoli alle richieste dei neuroni nella rete.

- **Strato Hidden**

Strato che processa l'elaborazione vera e propria, con possibilità che abbia una struttura con più livelli di neuroni.

- **Strato di Output**

Riceve i risultati delle elaborazioni precedenti fatte da H, che vengono adattati alle richieste del successivo livello-blocco della rete neurale.

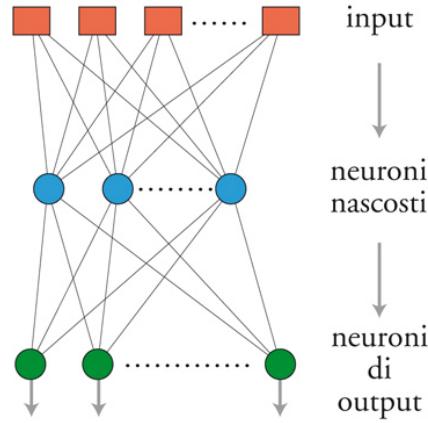


Figura 4.7: Strati Rete Neurale

A seconda della tipologia di apprendimento scelta, abbiamo diverse architetture generali. Si parte da una rete feedforward ad uno strato, con nodi in input e uno strato di output, che propaga il segnale in avanti in modo aciclico, partendo dal livello di input e terminando in quello di output. Le connessioni non tornano indietro e non ne esistono di trasversali nel livello di output[36].

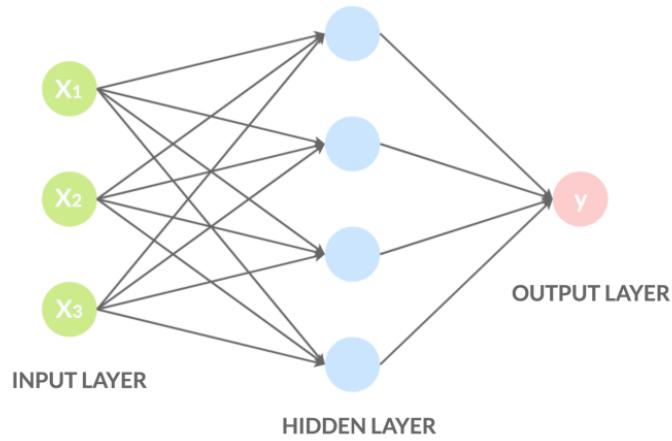


Figura 4.8: Rete feedforward ad uno strato

Le reti feedforward multistrato, invece, implementano nella propria architettura più strati di neuroni nascosti, dove per ognuno di essi si hanno connessioni entranti da quello precedente ed uscenti in quello successivo. Naturalmente, il

fenomeno influisce nell'iterazioni tra neuroni, maggiori in questa tipologia di architettura[36].

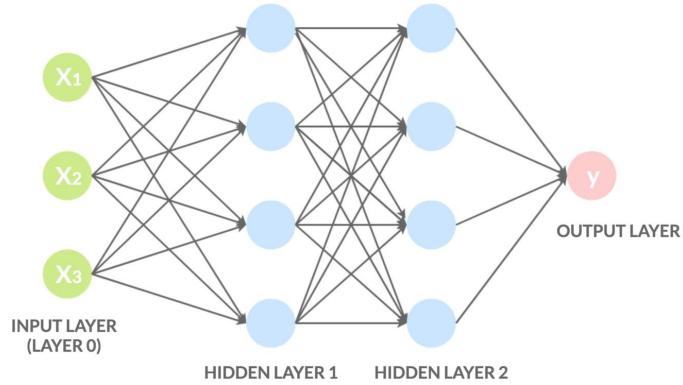


Figura 4.9: Rete feedforward ad uno multistrato

Ma come fa una rete neurale ad apprendere?

L'apprendimento è diviso in tre fasi principali:

- **Fase di associazione**

Possiamo avere due forme. La prima è l'autoassociazione, utilizzata nell'apprendimento non supervisionato, la quale memorizza un insieme di pattern in input e ne riceve come testing uno parziale o distorto, ai fini di ottenerne il completo e verificare la correttezza del training svolto.

La seconda è l'eteroassociazione, dove un pattern p_1 memorizzato opera uno stimolo per richiamare un altro pattern p_2 , con $p_1 \neq p_2$.

L'associazione è, dunque, divisa in due fasi: la prima di memorizzazione, la seconda di richiamo.

- **Fase di riconoscimento**

Questa fase, assegna i pattern alle varie classi che, in virtù della fase di addestramento precedente, sono in grado di riconoscere anche pattern appartenenti ad una stessa categoria e mai visti in fase di addestramento. Ogni pattern rappresenta un punto nello spazio decisionale multidimensionale, diviso in regioni associate ad una classe grazie alla fase di training.

Per quest'ultima, si possono utilizzare sia reti feedforward, sia una sistema diviso in due reti: una non supervisionata per l'estrazione delle features e una per la classificazione[36].

- **Fase di approssimazione delle funzioni**

Si consideri una funzione di mappatura f tale che $d = f(x)$, con x input e d output, candidato perfetto per l'apprendimento supervisionato. Si crei una rete neurale che approssima la funzione f , in modo che la funzione F descriva la mappa input-output realizzata e sia vicina ad f in senso euclideo, per tutti gli input forniti $(x_i, d_i)_{i=1}^n$. Si ottiene $\|F(x) - f(x)\| < \epsilon, \forall x$, dove ϵ è un numero piccolo positivo, che diminuisce in base all'aumento delle dimensioni e dei parametri liberi.

4.2 Algoritmi di Deep Learning

Il deep Learning è quel caso particolare di feature learning, a sua volta branca del machine learning, che caratterizza i processi di reti neurali artificiali dotate di due o più strati, spesso chiamate multilayed o a hidden layers; capaci di processare informazioni in modo non-lineare, in relazione alla particolare funzione di attivazione scelta[3].

L'approfondita attività di ricerca compiuta negli ultimi venti anni, unitamente alla disponibilità di unità di elaborazione grafica (GPU), in grado di supportare algoritmi paralleli su un numero elevatissimo di core, ha permesso la formulazione di nuove architetture per il deep learning.

Le più utilizzate sono[2]:

- **RNN**

Reti Neurali Ricorrenti, con loop tra i neuroni.

- **LSTM**

Reti Neurali a breve e lungo termine, tipi speciali di RNN.

- **CNN**

Reti Neurali Convoluzionali, tipo di rete neurale artificiale feed-forward, in cui il pattern di connettività tra i neuroni sono disposti in maniera tale da rispondere alle regioni di sovrapposizione, che tassellano il campo visivo (ispirandosi agli animali).

4.2.1 Reti Convoluzionali

La Convolutional Neural Networks (CNN) rappresenta un'architettura di rete neurale artificiale, di grande successo nelle applicazioni di computer vision, ampiamente utilizzata in ambiti di processamento relativo ai media, come audio, video e immagini.

Tramite una rete neurale convoluzionale, il computer è in grado di classificare un'immagine, identificando con buona probabilità il suo contenuto[9].

4.2.2 Reti CNN & Image Recognition

L'image recognition imita sostanzialmente il funzionamento del cervello umano, basandosi sulle cosiddette reti neurali convoluzionali (CNN), consentendo di classificare le immagini fisse o in movimento.

Il processo dell'image recognition, non è molto diverso da un modello di machine learning. Innanzitutto, dall'immagine viene estratto un gran numero di caratteristiche (features). Senza andare troppo nel dettaglio, una volta che ogni immagine è stata convertita in migliaia di features, si può iniziare ad addestrare un modello. Una volta fatto ciò, il modello può essere utilizzato per riconoscere o anche prevedere un'immagine sconosciuta[6].

4.2.3 Architettura CNN

Una rete di questo tipo, è composta da diversi strati. Troviamo:

- **Livello di input:** si passa il dataset, da analizzare, nel formato desiderato (immagine in scala RGB, in scala di grigi etc.).
- **Livello convoluzionale:** l'obiettivo, di questo livello, è individuare schemi (curve, angoli, circonferenze o quadrati). Il numero di questi

livelli, dovrebbe essere direttamente proporzionale alla complessità di calcolo, inerente alla caratteristica da ricercare nei dati.

- **Livello Rectified Linear Units:** semplicemente ReLU, questo livello annulla i valori negativi ottenuti nel passo precedente, applicando la funzione $f(x) = \max(0, x)$. Lo scopo è quello di introdurre la non linearità, in un sistema che sostanzialmente sta calcolando operazioni lineari durante i livelli convoluzionali (tramite il prodotto scalare tra il filtro e il campo ricettivo), permettendo una fase di apprendimento più veloce. Si deduce che il livello ReLU non influenza i campi ricettivi, relativi al livello convoluzionale[9].
- **Livello Pool:** si rileva se la caratteristica, oggetto di studio, sia presente nel livello precedente. Tutto questo, semplifica e rende più grezza l'immagine mantenendo gli attributi adoperati nel livello convoluzionale.
- **Livello Fully Connected:** Semplicemente FC, connette tutti i neuroni dello strato precedente, al fine di stabilire le varie classi identificative visualizzate nel layer antecedenti, secondo una determinata probabilità. Ogni classe, rappresenta una possibile risposta finale fornita in output.

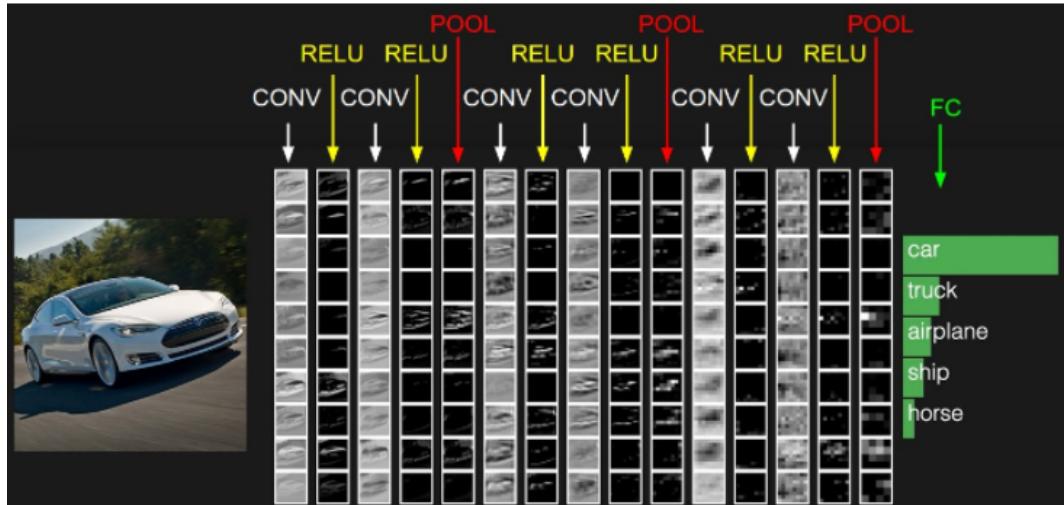
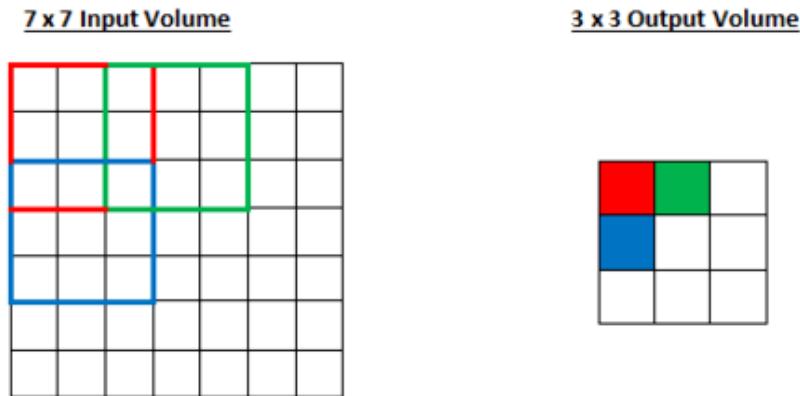


Figura 4.10: Rete CNN

Tipicamente le CNN ricevono in input dei dati bidimensionali, nel nostro caso le immagini. Avremo una o più matrici in input ed una chiamata kernel, che verrà traslata al fine di calcolarne il risultato ,chiamato feature map. Il passo di traslazione prende il nome di stride. Il risultato ottenuto in un livello convoluzionale, infatti, è ricavato tramite il prodotto scalare, tra i valori del filtro e i valori del livello, facendo spostare di un determinato stride verso destra, il campo ricettivo. L'insieme dei valori ottenuti forma la mappa di attivazione. Ripetendo questa procedura per ogni strato, se ne generano di differenti, costruendo una mappa per ogni livello analizzato.

Figura 4.11: Esempio di filtro 3×3 con stride 2

Un altro fattore rilevante, è rappresentato dallo zero-padding, strato da apporre al volume di input iniziale, al fine di evitare di perdere alcune informazioni dal passaggio da un layer all'altro[16]. Man mano che si continuano ad applicare i livelli convoluzionali, la dimensione del volume diminuisce più velocemente. Nei primi strati della rete, però, è bene conservare il maggior numero di informazioni possibili, relative all'input originale. In questo modo, si possono estrarre caratteristiche di basso livello, altrimenti perdute e non presenti in layer successivi[9]. Lo strato di zero-padding, quindi, riempie il volume di input con zeri attorno al bordo.

4.2.4 LeNet-5

La rete neurale LeNet-5, fu creata nel 1998 da Yann LeCun[25] e da allora viene sfruttata in domini applicativi diversi, come il riconoscimento di scrittura o di immagini, rigorosamente in scala di grigio.

L'implementazione originale, ha la seguente architettura[25]:

- **Livello di input:** le dimensioni delle immagini in input sono $32 \times 32 \times 1$.
- **Strato Convoluzionale 1:** formato da 6 filtri convoluzionali 5×5 con uno stride di 1 e padding 0, produce un sample $28 \times 28 \times 6$.
- **Strato di Pool 1:** l'average pooling conduce a un volume di $14 \times 14 \times 6$.
- **Strato Convoluzionale 2:** formato da 16 filtri convoluzionali 5×5 con uno stride di 1 e padding 0, produce un sample $10 \times 10 \times 6$.
- **Strato di Pool 2:** l'average pooling conduce a un volume di $5 \times 5 \times 6$.
- **Strato FC 1:** sfrutta l'algoritmo softmax collegando i 400 parametri precedenti a 256 neuroni.
- **Strato FC 2:** collega i 256 precedenti a 84 neuroni successivi.
- **Softmax:** collega gli 84 precedenti a 10 neuroni (classi attese).

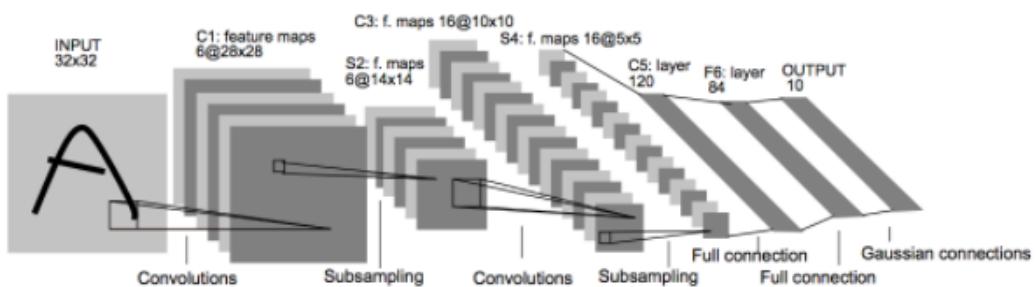


Figura 4.12: LeNet-5

Questa rete, è stata d'ispirazione alla rete convoluzionale scelta per la fase di sperimentazione, con ulteriori informazioni che saranno dettagliate in fase successiva.

4.3 Funzione di Loss

La funzione di perdita viene utilizzata per determinare l'errore tra l'output di un algoritmo e il valore target specificato. In parole povere, si esprime quanto lontano dalla previsione è l'output calcolato [11]. Esistono numerose tipologie di funzioni di loss, a seconda del problema in esame. Si riportano quelle specifiche per il problema di classificazione, oggetto di studio nei capitoli successivi:

- **Hinge Loss**

La Hinge Loss è una funzione di perdita utilizzata, insieme all'algoritmo Support Vector Machine (SVM), per problemi binari di classificazione. Ipotizzando un iperpiano lineare che separa due classi, si studia il segno della funzione z , per capire la posizione del data point

$z = \text{sign}(wx + w_0)$, con vettore dei pesi w , vettore delle caratteristiche x e bias w_0 . La funzione appena vista, conterrà valori positivi se si trova da una parte della linea, negativi se si troverà dall'altra. Il valore di segno z , rappresenta la distanza, positiva o negativa, dalla linea.[11]. Volendo minimizzare le classificazioni errate, si ha che un minimo di penalità sarà assegnata quando il dato ricade nel margine. Una penalità più elevata, invece, si otterrà quando la classificazione è errata.

Il tutto è descritto dalla formula $L_{\text{Hinge}} = \max(0; 1 - yz)$. In altre parole, quando la classificazione è corretta la perdita è nulla, in caso contrario si ha come perdita $1 - yz$, in quanto la classificazione è errata[11].

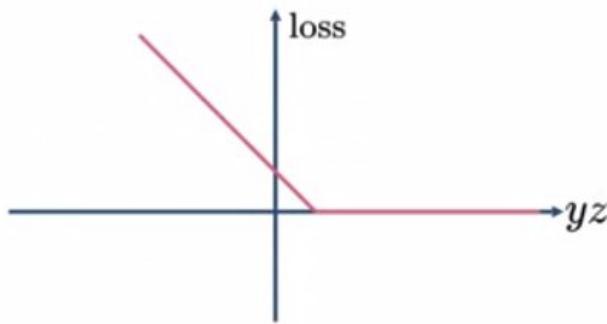


Figura 4.13: Hinge Loss

- **Cross Entropy Loss:** L'entropia incrociata è una misura della differenza tra due distribuzioni di probabilità, per una determinata variabile casuale o per un insieme di eventi[11]. Si riduce al minimo la differenza tra due probabilità: Q per la distribuzione attuale e P per l'approssimazione della distribuzione target. Si otterrà $H(x) = -\sum_{x=1}^X P(x) * \ln(Q(x))$, dove $P(x)$ è la probabilità dell'evento x in P , $Q(x)$ è la probabilità dell'evento x in Q . Questo calcolo, viene utilizzato per distribuzioni di probabilità discrete, mentre per quelle continue si utilizza l'integrale tra gli eventi, al posto della somma.

4.4 Metriche di classificazione utilizzate

4.4.1 Matrice di Confusione

Una matrice di confusione, nota anche come matrice di errore, è un layout di tabella specifico tramite il quale si consente la visualizzazione delle prestazioni di un algoritmo, tipicamente di apprendimento supervisionato[1].

		ACTUAL	
		Positive	Negative
PREDICTED	Positive	TRUE POSITIVE	FALSE POSITIVE
	Negative	FALSE NEGATIVE	TRUE NEGATIVE

Figura 4.14: Matrice di confusione

Per comprendere la matrice di confusione, si consideri un problema di classificazione a due classi, con i due risultati "Positivo" e "Negativo". Dato un punto dati da prevedere, il risultato del modello sarà uno qualsiasi di questi due. La matrice è utile a tenere traccia delle seguenti misure:

- **True Positives (TP)**: punti identificati correttamente come positivi.
- **True Negatives (TN)**: punti negativi identificati correttamente come negativi.
- **Falsi positivi (FP)**: punti negativi erroneamente identificati come positivi.
- **Falsi negativi (FN)**: punti positivi erroneamente identificati come negativi.

L'obiettivo della valutazione di un modello utilizzando la matrice di confusione, è in definitiva massimizzare i valori di TP e TN e minimizzare i valori di FP e FN.

4.4.2 Precisione, Richiamo ed F1-Score

La precisione è la frazione di istanze rilevanti tra le istanze recuperate, mentre il richiamo è la frazione della quantità totale di istanze rilevanti effettivamente recuperate. Sia la precisione che il richiamo si basano su comprensione e misura della pertinenza[1].

$$\text{PRECISION} = \frac{\text{TRUE POSITIVES (TP)}}{\text{TRUE POSITIVES + FALSE POSITIVES (TP) (FP)}}$$

Figura 4.15: Precision

$$\text{RECALL} = \frac{\text{TRUE POSITIVES (TP)}}{\text{TRUE POSITIVES + FALSE NEGATIVES (TP) (FN)}}$$

Figura 4.16: Recall

Lo score F1 è una misura dell'accuratezza di un test, calcolato come media armonica o media ponderata di precisione e richiamo.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Figura 4.17: F1-score

Tramite la matrice di confusione e le metriche sovrastanti, si valutano le performance ottenute, controllando fenomeni indesiderati come l'overfitting di un modello.

4.4.3 Curva di ROC

La curva ROC (Receiver Operating Characteristic) è un grafico che mette in relazione la sensibilità e la specificità di un test diagnostico, al variare del valore di cut-off (valore soglia). L'analisi della curva ROC, permette di valutare l'accuratezza, di determinare il valore di cut-off più appropriato e di confrontare le performance di due, o più, diversi test[15]. Le curva è ottenuta tramite due indicatori:

- **Sensibilità (Recall)**: misura uguale a $TP/(TP+FN)$, come spiegato in precedenza.
- **Specificità**: misura uguale a $TN/(TN+FP)$.

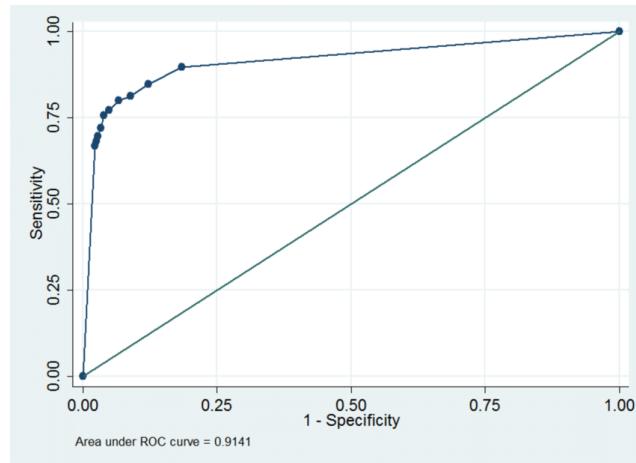


Figura 4.18: Curva di ROC

L'area al di sotto della curva di ROC (AUC) può essere compresa tra [0,1]. Se $AUC < 0.5$, indica un test non informativo, se $AUC > 0.5$ indica un test che via via si avvicina ad essere più accurato. Il test con l'AUC maggiore è, generalmente, il migliore[15]. Nella sperimentazione progettuale, questo indicatore sarà utilizzato per mostrare il comportamento del miglior modello nel rilevamento di anomalie.

Capitolo 5

Business Forecasting

5.1 Metodologia sperimentale adottata

L'intelligenza artificiale non ha impiegato troppo tempo per uscire dagli ambienti accademici ed entrare nell'orbita del mercato e del business, mostrando una concretezza di applicazioni che stanno via via contribuendo alla divulgazione, tra le aziende, di una cultura basata sull'IA e più in generale sullo sfruttamento intelligente dei dati. L'utilizzo di algoritmi di machine & deep learning consente di elaborare e analizzare i dati di origine visiva, trasformandoli, tramite un opportuno preprocessing, in informazioni utili e codificabili da una macchina.

Alla luce di questo, la soluzione che si propone aggrega in un'immagine tutte le informazioni ricavate da ogni singolo KPI aziendale (valori estratti da procedure implementate nel sistema EVO-BI), per un lasso temporale stabilito.

In particolare, ogni colonna rappresenterà un indicatore scelto (con valore settimanale, bisettimanale o mensile) visualizzando i risultati su una finestra temporale raffigurante, ad esempio, un andamento mensile, bimestrale, trimestrale etc.

Al termine di tutte le fasi di processamento dei dati, dettagliate nelle sezioni successive, si otterrà un dataset composto da svariate immagini in scala di grigio, dove ogni pixel ha un valore di luminosità compreso tra 0 (nero) e 255

(bianco). Maggiore sarà la luminosità raggiunta da ogni elemento relativo ad una colonna, migliore sarà l'andamento del KPI per quella stessa colonna. I dati così aggregati permetteranno ad un calcolatore di capire, in modo abbastanza chiaro e completo, le performance ottenute nella finestra temporale considerata.

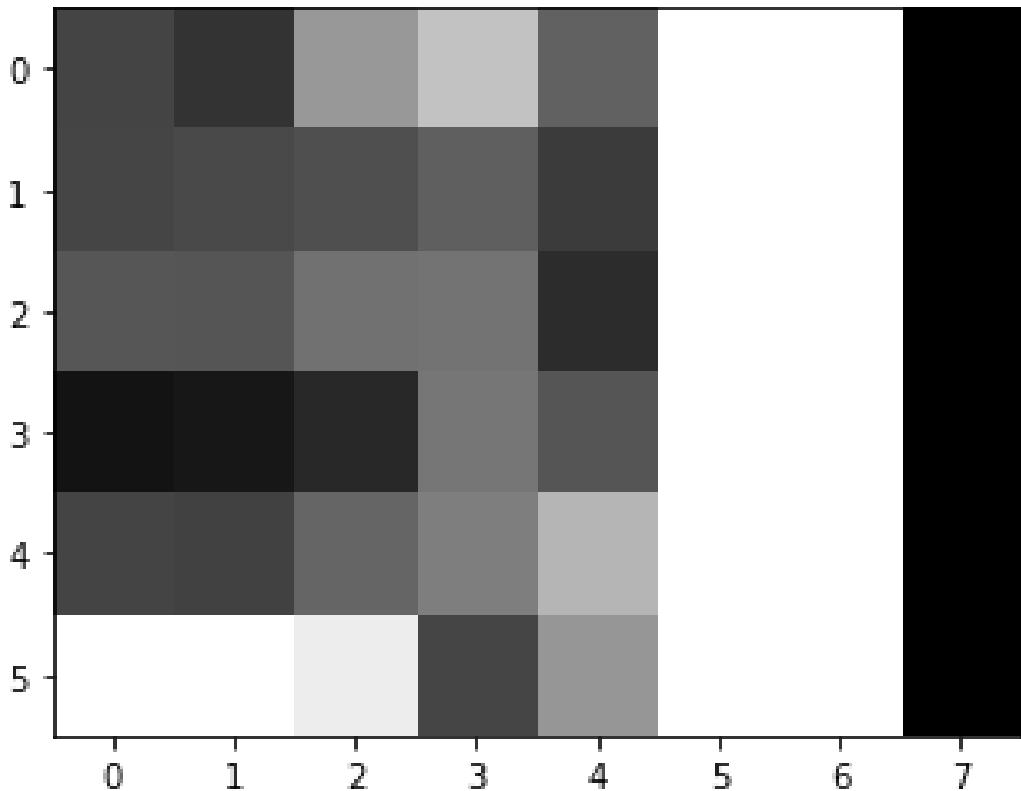


Figura 5.1: Trimestre con KPI a 15 giorni

Ma come si effettua tutto questo?

Un dataset di questo tipo, può sfruttare le tecniche di classificazione di immagini tipiche del Machine Learning, prestandosi ancora meglio all'utilizzo di reti CNN.

In particolare, l'utilizzo di questi modelli trova grande applicazione nell'ambito dell'Image Recognition. In virtù di tutte le politiche adottate nella costruzione del dataset e nel preprocessing delle immagini che lo compongono, si riesce ad

estrarre un gran numero di caratteristiche, permettendo una facile identificazione di quelle chiavi che differenziano le diverse tipologie di immagini, presenti nel dataset. Una volta fatto ciò, si può iniziare ad addestrare un modello sui dati storici di cui si è in possesso. Terminata questa fase, se ne ottengono diversi, uno per ogni classificatore o rete neurale implementato.

Dopo un'accurata fase di analisi, tramite le metriche adottate, si sceglierà il modello migliore in questo contesto applicativo. Così facendo, si può ottenere una previsione affidabile, sull'andamento aziendale, per nuovi dati forniti in input e nello stesso formato di quelli iniziali valutando, ad esempio, il superamento di determinati obiettivi preposti e potendo, inoltre, rilevare facilmente eventuali anomalie.

5.2 Creazione del Dataset

Durante la collaborazione con EVO-BI, azienda spin off dell'Università della Calabria, è stato possibile analizzare lo storico triennale di un cliente, relativo agli anni 2018/2019/2020, anonimizzando e offuscando opportunamente il tutto. Dopo un'attenta analisi, sui vari dati disponibili, si è scelto di utilizzare per la creazione del dataset da esaminare, i seguenti indicatori di performance:

- **Venduto:** valore delle vendite.
- **Costo:** costo attribuibile alla produzione di beni e/o servizi venduti.
- **Margine:** margine di profitto (Venduto-Costo), espresso sia in valore che in percentuale.
- **Trend delle spedizioni:** andamento della spedizione dei prodotti nel periodo scelto.
- **Margine % sulle spedizioni:** margine di profitto, derivante dal trend delle spedizioni.

Si è scelto, inoltre, di includere altre due colonne come indicatori esterni: andamento di consumi e prezzi, nel lasso temporale considerato, non estraibili direttamente dal contesto aziendale ma aventi un impatto notevole in termini di performance.

I dati, relativi ad ogni KPI sopra indicato, sono stati estratti con finestre a 7,15 e 30 giorni, tramite script sql di mia creazione.

Un indicatore, con finestra settimanale su un anno scelto, sarà costituito da 48 righe, ognuna indicante la performance aziendale nella settimana in esame.

ID	costoS2018
0	1 173611.55909
1	2 238598.94887
2	3 223596.04487
3	4 235867.28252
4	5 234566.82341
5	6 249100.81388
6	7 201718.83164
7	8 151963.49538
8	9 198929.59171
9	10 245025.94749
10	11 257918.37380
11	12 583797.54032
12	13 93970.03810
13	14 218326.58048
14	15 214985.63912
15	16 180999.38509
16	17 196455.21080
17	18 193315.99630
18	19 172532.23876
19	20 191069.15435
20	21 172768.42413
21	22 187449.62634
22	23 161265.33350
23	24 167344.33771
24	25 188064.55074
25	26 251805.78665
26	27 227822.65546
27	28 217724.03799
28	29 257961.68351
29	30 355477.63862
30	31 259641.76786
31	32 236712.66222
32	33 195412.19150
33	34 168382.15469
34	35 249582.39878
35	36 168698.25120
36	37 181803.43026
37	38 229039.83683
38	39 225881.52867
39	40 253480.74969
40	41 188551.26718
41	42 221585.71357
42	43 159646.96220
43	44 144217.92713
44	45 195352.07867
45	46 158245.61713
46	47 411755.02585
47	48 150054.02032

Figura 5.2: KPI costo con finestra settimanale per l'anno 2018

Si precisa che, la riga 0 corrisponde alla prima settimana di Gennaio, la riga 1 alla seconda e così via fino all'ultima settimana del mese di Dicembre, relativa all'anno scelto.

La seconda e la terza tabella sottosanti, invece, rappresentano l'indicatore costo con visualizzazione a 15 o 30 giorni, aventi rispettivamente 24 e 12 righe. In questo caso, l'indice 0 indica il costo relativo alle prime due settimane di Gennaio, oppure il valore dell'intero mese, l'indice 1 il valore della terza e quarta settimana di Gennaio, oppure il valore del mese di Febbraio etc.

ID	costoSS2018
0	1 412210.50796
1	2 459463.32739
2	3 483667.63729
3	4 353682.32702
4	5 443955.53920
5	6 841715.91412
6	7 312296.61858
7	8 395985.02421
8	9 389771.20710
9	10 363601.39311
10	11 360218.05047
11	12 328609.67121
12	13 439870.33739
13	14 445546.69345
14	15 613439.32213
15	16 496354.43008
16	17 363794.34619
17	18 418280.64998
18	19 410843.26709
19	20 479362.27836
20	21 410136.98075
21	22 303864.88933
22	23 353597.69580
23	24 561809.04617

Figura 5.3: KPI costo con finestra a 15 giorni per l'anno 2018

	ID	costoM2018
0	1	8.716738e+05
1	2	8.373500e+05
2	3	1.285671e+06
3	4	7.082816e+05
4	5	7.533726e+05
5	6	6.888277e+05
6	7	8.854170e+05
7	8	1.109794e+06
8	9	7.820750e+05
9	10	8.902055e+05
10	11	7.140019e+05
11	12	9.154067e+05

Figura 5.4: KPI costo con finestra a 30 giorni per l'anno 2018

5.2.1 Aggregazione di KPI e preprocessing dei dati

La piattaforma di sviluppo di scelta è Google Colab Pro. A fronte di una piccola quota mensile si può accedere, grazie al linguaggio Python e alla sua pacchettizzazione, a tutto il necessario per affrontare un problema di machine & deep Learning sfruttando, inoltre, GPU molto performanti da remoto.

Si prenda ora, come esempio per questa fase, l'anno 2018.

La prima operazione effettuata, consiste nell'aggregare i KPI relativi alla stessa visualizzazione, ottenendo una matrice di rappresentazione per ogni anno analizzato, rispettivamente a 7,15 e 30 giorni. Inoltre, viene creata la colonna relativa al margine non in %, sottraendo la colonna Venduto a quella Costo.

```

tab7gg2018=pd.merge(tabelle7gg2018[0],tabelle7gg2018[1])
for i in range(2,len(tabelle7gg2018)):
    tab7gg2018=pd.merge(tab7gg2018,tabelle7gg2018[i])
tab7gg2018 = tab7gg2018.assign(margineS2018 =
                                tab7gg2018['vendutoS2018'] - tab7gg2018['costoS2018'])
tab7gg2018.drop('ID',axis=1,inplace=True)

```

	costoS2018	marginePercSpeditoMobileAverageS2018	percMargineS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	vendutoS2018	margineS2018
0	173611.55909	43.777438	0.382267		1.0 281046.347650	107434.788560
1	238596.94887	30.462190	0.275797		1.0 329464.322985	90865.374115
2	223596.04487	25.553304	0.252653		1.0 299186.495240	75590.450370
3	235867.28252	27.740722	0.246410		1.0 312991.570481	77124.287961
4	234566.82341	23.689413	0.248969		1.0 312326.715490	77759.892080
5	249100.81388	26.732053	0.280089		1.0 346016.533371	96915.719491
6	201718.83164	27.857213	0.244377		1.0 266957.057600	65238.225960
7	151963.49538	29.850659	0.295689		1.0 215762.016498	63798.521118
8	198929.59171	33.357287	0.265463		1.0 270823.439880	71893.848170
9	245025.94749	36.333872	0.279703		1.0 340173.542522	95147.595032
10	257918.37380	28.319456	0.280125		1.0 358282.379829	100364.006029
11	583797.54032	34.161169	0.206090		1.0 735345.239106	151547.698786
12	93970.03810	31.344172	0.302178		1.0 134661.986422	40691.948322
13	218326.58048	33.272496	0.269500		1.0 298872.857463	80546.276983
14	214985.63912	31.392768	0.286369		1.0 301256.174507	86270.535387
15	180999.38509	38.202497	0.296449		1.0 257265.679606	76266.294516
16	196455.21080	33.408727	0.280779		1.0 273150.234750	76695.023950
17	193315.99630	39.822346	0.294043		1.0 273835.407193	80519.410893
18	172532.23876	34.934880	0.407819		1.0 291350.985650	118818.746890
19	191069.15435	35.940673	0.274955		1.0 263527.641113	72458.486763
20	172768.42413	33.410997	0.318650		1.0 253567.823830	80799.399700
21	187449.62634	36.120497	0.391218		1.0 307909.619247	120459.992907
22	161265.33350	39.686475	0.387905		1.0 263464.753094	102199.419594
23	167344.33771	41.546264	0.255788		1.0 224861.256900	57516.919190
24	188064.55074	43.870635	0.348693		1.0 288749.796971	100685.246231
25	251805.78665	38.330832	0.293799		1.0 356564.164104	104758.377454
26	227822.65546	31.104891	0.370474		1.0 361895.837712	134073.182252
27	217724.03799	36.195199	0.371316		1.0 346317.576726	128593.538736
28	257961.68351	35.263928	0.291806		1.0 364253.169255	106291.485745
29	355477.63862	28.208319	0.256647		1.0 478208.912274	122731.273654
30	259641.76786	29.774851	0.254845		1.0 348440.230962	88798.463102
31	236712.66222	27.714314	0.271173		1.0 324786.122480	88073.460260
32	195412.19150	24.314925	0.243091		1.0 258171.331437	62759.139937
33	168382.15469	35.597011	0.329534		1.0 251142.040299	82759.885609
34	249582.39878	38.791161	0.265033		1.0 339583.503295	90001.104515
35	168698.25120	31.842001	0.356843		1.0 262297.548351	93599.297151
36	181803.43026	44.910289	0.347559		1.0 278651.468966	96848.038706
37	229039.83683	32.591635	0.240081		1.0 301400.466672	72360.629842
38	225881.52867	26.462473	0.261550		1.0 305886.043816	80004.515146
39	253480.74969	23.539279	0.084141		1.0 276768.540916	23287.791226
40	188551.26718	34.111591	0.267954		1.0 257567.630119	69016.362939
41	221585.71357	32.369432	0.294923		1.0 314271.738955	92686.025385
42	159646.96220	38.126862	0.340105		1.0 241928.143788	82281.181588
43	144217.92713	33.484042	0.293932		1.0 204255.028484	60037.101354
44	195352.07867	32.744505	0.266270		1.0 266245.196187	70893.117517
45	158245.61713	35.051285	0.334358		1.0 237734.007271	79488.390141
46	411755.02585	43.920009	0.225167		1.0 531411.587257	119656.561407
47	150054.02032	34.075926	0.454393		1.0 275022.503505	124968.483185

Figura 5.5: Unione KPI con finestra settimanale 2018

Successivamente, si passa ad una normalizzazione dei valori relativi ad ogni colonna, tramite l'uso del MinMaxScaler().

La colonna relativa al trend, invece, passerà alla rappresentazione di valori negativi con 0, mentre 1 sarà utilizzato per i positivi.

```
column_names_to_normalize = [ 'costoS2018' ,
    'marginePercSpeditoMobileAverageS2018' , 'percMargineS2018' ,
    'vendutoS2018' , 'margineS2018' ]
x = tab7gg2018 [ column_names_to_normalize ]. values
x_scaled = MinMaxScaler () . fit _ transform ( x )
df_temp = pd . DataFrame ( x_scaled , columns=column_names_to_normalize ,
                           index = tab7gg2018 . index )
tab7gg2018 [ column_names_to_normalize ] = df_temp
x=tab7gg2018 [ 'trendMarginePercSpeditoByCompanyAndCostTypeS2018' ]. values
for i in range ( len ( x )):
    if ( x [ i ] == -1 ):
        x [ i ] = 0
tab7gg2018 [ 'trendMarginePercSpeditoByCompanyAndCostTypeS2018' ] = x
```

	costoS2018	marginPercSpeditoMobileAverageS2018	percMarginoS2018	trendMarginPercSpeditoByCompanyAndCostTypeS2018	vendutoS2018	marginoS2018
0	0.162591	0.946991	0.805198		1.0	0.243696
1	0.295265	0.323939	0.517637		1.0	0.324301
2	0.264636	0.094241	0.455128		1.0	0.273896
3	0.289688	0.196595	0.438266		1.0	0.296878
4	0.287033	0.007025	0.445178		1.0	0.295771
5	0.316705	0.149397	0.529229		1.0	0.351857
6	0.219973	0.202046	0.432776		1.0	0.220241
7	0.118396	0.281287	0.571362		1.0	0.135013
8	0.214279	0.459408	0.489726		1.0	0.226678
9	0.308386	0.598689	0.528186		1.0	0.342130
10	0.334706	0.223676	0.529326		1.0	0.372277
11	1.000000	0.497023	0.329368		1.0	1.000000
12	0.000000	0.365209	0.588888		1.0	0.000000
13	0.253878	0.455440	0.500629		1.0	0.273373
14	0.247058	0.367483	0.546190		1.0	0.277341
15	0.177673	0.686127	0.573415		1.0	0.204107
16	0.209227	0.461815	0.531092		1.0	0.230551
17	0.202818	0.761923	0.566917		1.0	0.231692
18	0.160387	0.533227	0.874210		1.0	0.260851
19	0.198231	0.561574	0.515363		1.0	0.214532
20	0.160870	0.461921	0.633377		1.0	0.197951
21	0.190842	0.588705	0.829373		1.0	0.288418
22	0.137386	0.755565	0.820425		1.0	0.214427
23	0.149796	0.842589	0.463595		1.0	0.150161
24	0.192097	0.951352	0.714519		1.0	0.256521
25	0.322227	0.692132	0.566258		1.0	0.369416
26	0.273265	0.354013	0.773346		1.0	0.378292
27	0.252648	0.592200	0.775620		1.0	0.352358
28	0.334795	0.548624	0.560875		1.0	0.382217
29	0.533877	0.218475	0.465915		1.0	0.571927
30	0.338225	0.291777	0.461048		1.0	0.355892
31	0.291414	0.195360	0.505148		1.0	0.316513
32	0.207098	0.036294	0.429302		1.0	0.205615
33	0.151915	0.564210	0.662773		1.0	0.193913
34	0.317688	0.713672	0.488565		1.0	0.341147
35	0.152560	0.388504	0.736531		1.0	0.212484
36	0.179315	1.000000	0.711456		1.0	0.239710
37	0.275750	0.423581	0.421173		1.0	0.277581
38	0.269302	0.136783	0.479157		1.0	0.285049
39	0.325647	0.000000	0.000000		1.0	0.236575
40	0.193091	0.494703	0.496454		1.0	0.204610
41	0.260532	0.413184	0.569293		1.0	0.299009
42	0.134082	0.682587	0.691324		1.0	0.178574
43	0.102583	0.465339	0.566617		1.0	0.115856
44	0.206975	0.430734	0.491906		1.0	0.219056
45	0.131221	0.538674	0.675802		1.0	0.171591
46	0.648769	0.953662	0.380892		1.0	0.660497
47	0.114497	0.493035	1.000000		1.0	0.233668

Figura 5.6: normalizzazione KPI con finestra settimanale 2018

In virtù della visualizzazione in scala di grigi, si portano tutti i valori nel range 0-255. Infine, si passa al riordinamento delle colonne, ottenendo la visualizzazione dell'anno considerato con finestra settimanale, che sarà salvato in una tabella csv apposita.

```
tab7gg2018=tab7gg2018*255
tab7gg2018=tab7gg2018 . astype( int )
columnsTitles = [ 'vendutoS2018' , 'costoS2018' , 'margineS2018' ,
                  'percMargineS2018' ,
                  'marginePercSpeditoMobileAverageS2018' ,
                  'trendMarginePercSpeditoByCompanyAndCostTypeS2018' ]
tab7gg2018 = tab7gg2018 . reindex( columns=columnsTitles )
tab7gg2018 . to_csv( './tabelleRicavate/tab7gg2018' , index=False )
tab7gg2018=pd . read_csv( './tabelleRicavate/tab7gg2018' )
```

	costoS2018	marginPercSpeditoMobileAverageS2018	percMarginS2018	trendMarginPercSpeditoByCompanyAndCostTypeS2018	vendutoS2018	marginS2018
0	41	241	205		255	62
1	75	82	131		255	82
2	67	24	116		255	69
3	73	50	111		255	75
4	73	1	113		255	75
5	80	38	134		255	89
6	56	51	110		255	56
7	30	71	145		255	34
8	54	117	124		255	57
9	78	152	134		255	87
10	85	57	134		255	94
11	255	126	83		255	255
12	0	93	150		255	0
13	64	116	127		255	69
14	62	93	139		255	70
15	45	174	146		255	52
16	53	117	135		255	58
17	51	194	144		255	59
18	40	135	222		255	66
19	50	143	131		255	54
20	41	117	161		255	50
21	48	150	211		255	73
22	35	192	209		255	54
23	38	214	118		255	38
24	48	242	182		255	65
25	82	176	144		255	94
26	69	90	197		255	96
27	64	151	197		255	89
28	85	139	143		255	97
29	136	55	118		255	145
30	86	74	117		255	90
31	74	49	128		255	80
32	52	9	109		255	52
33	38	143	169		255	49
34	81	181	124		255	86
35	38	99	187		255	54
36	45	255	181		255	61
37	70	108	107		255	70
38	68	34	122		255	72
39	83	0	0		255	60
40	49	126	126		255	52
41	66	105	145		255	76
42	34	174	176		255	45
43	26	118	144		255	29
44	52	109	125		255	55
45	33	137	172		255	43
46	165	243	97		255	168
47	29	125	255		255	59

Figura 5.7: Tabella con finestra settimanale anno 2018

Per i medesimi indicatori con KPI a visualizzazione più ampia, 15 e 30 giorni, si otterrà:

	vendutoSS2018	costoSS2018	marginSS2018	percMargineSS2018	marginPercSpeditoMobileAverageSS2018	trendMarginePercSpeditoByCompanyAndCostTypeSS2018	
0	68	51	152	194	98		255
1	69	73	79	95	59		255
2	86	85	114	115	45		255
3	19	23	41	118	85		255
4	68	66	101	126	181		255
5	255	255	237	69	150		255
6	0	3	28	134	137		255
7	48	43	94	149	207		255
8	43	40	86	145	230		255
9	46	28	140	220	169		255
10	49	26	156	238	178		255
11	21	11	90	197	255		255
12	81	64	163	185	209		255
13	106	67	255	254	179		255
14	157	146	201	124	66		255
15	92	91	117	112	59		255
16	29	28	67	142	170		255
17	65	54	128	168	117		255
18	56	50	105	150	128		255
19	57	83	0	0	0		255
20	53	50	93	138	125		255
21	4	0	62	186	140		255
22	27	23	75	159	163		255
23	144	122	226	166	149		255

Figura 5.8: normalizzazione KPI con finestra 15 giorni 2018

	vendutoM2018	costoM2018	marginM2018	percMargineM2018	marginPercSpeditoMobileAverageM2018	trendMarginePercSpeditoByCompanyAndCostTypeM2018	
0	82	78	102	120	59		255
1	53	63	40	72	85		255
2	255	255	190	26	150		255
3	0	8	14	118	207		255
4	39	27	99	187	169		255
5	20	0	115	250	255		255
6	129	83	254	255	179		255
7	187	179	173	76	59		255
8	42	39	73	141	117		255
9	61	86	0	0	0		255
10	9	10	41	147	140		255
11	113	96	159	153	149		255

Figura 5.9: normalizzazione KPI con finestra 30 giorni 2018

Tutte queste operazioni, vengono applicate anche alle annate 2019 e 2020, portando ad ottenere una rappresentazione di ogni anno con finestre a 7,15

e 30 giorni. Il tutto, come sarà mostrato nella sezione successiva, genererà il dataset da analizzare.

5.2.2 Creazione delle immagini

Si passa ora alla creazione delle singole immagini che comporranno il dataset, da utilizzare nelle fasi successive di analisi. Si prenda, come esempio, la rappresentazione dell'anno 2018 mostrata precedentemente.

```
def genera(t , indice , limite ):  
    tabs = []  
    i=0  
    j=indice  
    while i<limite :  
        tabs . append( t . iloc [ i :j , : ] )  
        i=j  
        j=j+indice  
    return tabs
```

Da questa, otterremo diverse finestre temporali: finestre ad una settimana composte da una riga, finestre a due settimane composte da due righe contigue, finestre mensili composte da quattro righe contigue e così via, producendo, inoltre, una rappresentazione bimestrale, trimestrale, quadrimestrale, semestrale, nonimestrale e annuale relativa all'anno in esame.

	vendutoS2018	costoS2018	margineS2018	percMargineS2018	marginePercSpeditoMobileAverageS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	
0	62	41	167	205	241		255

Figura 5.10: Rappresentazione di una settimana

	vendutoS2018	costoS2018	margineS2018	percMargineS2018	marginePercSpeditoMobileAverageS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	
0	62	41	167	205	241		255
1	82	75	134	131	82		255

Figura 5.11: Rappresentazione di due settimane con KPI a 7gg

	vendutoS2018	costoS2018	margineS2018	percMargineS2018	marginePercSpeditoMobileAverageS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	
0	62	41	167	205	241		255
1	82	75	134	131	82		255
2	69	67	103	116	24		255
3	75	73	107	111	50		255

Figura 5.12: Rappresentazione di un mese con KPI a 7gg

	vendutoS2018	costoS2018	margineS2018	percMargineS2018	marginePercSpeditoMobileAverageS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	
0	62	41	167	205	241		255
1	82	75	134	131	82		255
2	69	67	103	116	24		255
3	75	73	107	111	50		255
4	75	73	108	113	1		255
5	89	80	146	134	38		255
6	56	56	83	110	51		255
7	34	30	80	145	71		255

Figura 5.13: Rappresentazione di un bimestre con KPI a 7gg

	vendutoS2018	costoS2018	marginoS2018	percMargineS2018	marginPercSpeditoMobileAverageS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	
0	62	41	167	205	241		255
1	82	75	134	131	82		255
2	69	67	103	116	24		255
3	75	73	107	111	50		255
4	75	73	108	113	1		255
5	89	80	146	134	38		255
6	56	56	83	110	51		255
7	34	30	80	145	71		255
8	57	54	96	124	117		255
9	87	78	142	134	152		255
10	94	85	153	134	57		255
11	255	255	255	83	126		255

Figura 5.14: Rappresentazione di un trimestre con KPI a 7gg

	vendutoS2018	costoS2018	marginoS2018	percMargineS2018	marginPercSpeditoMobileAverageS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	
0	62	41	167	205	241		255
1	82	75	134	131	82		255
2	69	67	103	116	24		255
3	75	73	107	111	50		255
4	75	73	108	113	1		255
5	89	80	146	134	38		255
6	56	56	83	110	51		255
7	34	30	80	145	71		255
8	57	54	96	124	117		255
9	87	78	142	134	152		255
10	94	85	153	134	57		255
11	255	255	255	83	126		255
12	0	0	34	150	93		255
13	69	64	113	127	116		255
14	70	62	125	139	93		255
15	52	45	105	146	174		255

Figura 5.15: Rappresentazione di un quadrimestre con KPI a 7gg

	venduto\$2018	costo\$2018	margin\$2018	percMargines\$2018	marginPercSpeditoMobileAverage\$2018	trendMarginePercSpeditoByCompanyAndCostType\$2018	
0	62	41	167	205	241		255
1	82	75	134	131	82		255
2	69	67	103	116	24		255
3	75	73	107	111	50		255
4	75	73	108	113	1		255
5	89	80	146	134	38		255
6	56	56	83	110	51		255
7	34	30	80	145	71		255
8	57	54	96	124	117		255
9	87	78	142	134	152		255
10	94	85	153	134	57		255
11	255	255	255	83	126		255
12	0	0	34	150	93		255
13	69	64	113	127	116		255
14	70	62	125	139	93		255
15	52	45	105	146	174		255
16	58	53	106	135	117		255
17	59	51	113	144	194		255
18	66	40	189	222	135		255
19	54	50	97	131	143		255
20	50	41	114	161	117		255
21	73	48	193	211	150		255
22	54	35	156	209	192		255
23	38	38	68	118	214		255

Figura 5.16: Rappresentazione di un semestre con KPI a 7gg

	venduto\$2018	costo\$2018	margin\$2018	percMargin\$2018	marginPercSpeditoMobileAverage\$2018	trendMarginPercSpeditoByCompanyAndCostType\$2018
0	62	41	167	205	241	255
1	82	75	134	131	82	255
2	69	67	103	116	24	255
3	75	73	107	111	50	255
4	75	73	108	113	1	255
5	89	80	146	134	38	255
6	56	56	83	110	51	255
7	34	30	80	145	71	255
8	57	54	96	124	117	255
9	87	78	142	134	152	255
10	94	85	153	134	57	255
11	255	255	255	83	126	255
12	0	0	34	150	93	255
13	69	64	113	127	116	255
14	70	62	125	139	93	255
15	52	45	105	146	174	255
16	58	53	106	135	117	255
17	59	51	113	144	194	255
18	66	40	189	222	135	255
19	54	50	97	131	143	255
20	50	41	114	161	117	255
21	73	48	193	211	150	255
22	54	35	156	209	192	255
23	38	38	68	118	214	255
24	65	48	153	182	242	255
25	94	82	161	144	176	255
26	96	69	220	197	90	255
27	89	64	209	197	151	255
28	97	85	165	143	139	255
29	145	136	197	118	55	255
30	90	86	130	117	74	255
31	80	74	128	128	49	255
32	52	52	78	109	9	255
33	49	38	118	169	143	255
34	86	81	132	124	181	255
35	54	38	139	187	99	255

Figura 5.17: Rappresentazione di un nonimestre con KPI a 7gg

E' bene precisare che, passando alla visualizzazione di KPI con valori a 15 giorni, una riga rappresenterà il valore di 2 settimane, 2 righe continue un mese e così via. Un discorso analologo vale per KPI a 30 giorni, dove una riga rappresenta un mese, due righe contigue un bimestre etc. Si mostrano un paio di esempi, utili a comprendere meglio quanto appena detto. Applicando la metodologia anche alle annate 2019 e 2020, si formerà il dataset completo da analizzare.

	vendutoSS2018	costoSS2018	marginSS2018	percMargineSS2018	marginPercSpeditoMobileAverageSS2018	trendMarginePercSpeditoByCompanyAndCostTypeSS2018	
0	68	51	152	194		98	255
1	69	73	79	95		59	

Figura 5.18: Rappresentazione di un mese con KPI a 15gg

	vendutoM2018	costoM2018	marginM2018	percMargineM2018	marginPercSpeditoMobileAverageM2018	trendMarginePercSpeditoByCompanyAndCostTypeM2018	
0	82	78	102	120		59	255

Figura 5.19: Rappresentazione di un mese con KPI a 30gg

5.2.3 Aggiunta vista andamento prezzi e consumi

Verranno ora aggiunte due colonne. Grazie ad i report su prezzi e consumi di prodotti agricoli, generati ogni trimestre da Ismea [7], si aggiungono le informazioni di contesto utili esterne, poichè i dati analizzati appartengono ad un'azienda che opera nel settore alimentare. I dati ricavati da tutti questi report sono stati tradotti in due tabelle csv, una relativa ai consumi, una relativa ai prezzi. Si specifica che, essendo report generati su trimestre, uno stesso valore viene propagato più volte per determinate finestre.

	consumi2018	consumi2019	consumi2020
0	255	128	255
1	128	255	255
2	128	0	128
3	128	0	255

Figura 5.20: Andamento Consumi

	prezzi2018	prezzi2019	prezzi2020
0	0	0	128
1	0	0	0
2	0	0	128
3	128	128	128

Figura 5.21: Andamento Prezzi

Per questi dati, vale la stessa chiave di lettura tenuta sugli indicatori KPI. Un cella più scura, a valore 0, indicherà una crescita dei prezzi della materia prima o un trend di consumi generali negativo. Viceversa un colore bianco 255, indicherà un ribasso del costo della materia prima o un aumento del trend dei consumi generale. Inoltre, un valore di stabilità per queste due colonne, verrà indicato con 128. Si mostra, a titolo di esempio, la visualizzazione di un quadri mestre con KPI a 7 giorni, relativo all'anno 2018.

	vendutoS2018	costoS2018	margineS2018	percMargineS2018	marginePercSpeditoMobileAverageS2018	trendMarginePercSpeditoByCompanyAndCostTypeS2018	consumi2018	prezzi2018	
0	62	41	167	205	241		255	255	0
1	82	75	134	131	82		255	255	0
2	69	67	103	116	24		255	255	0
3	75	73	107	111	50		255	255	0
4	75	73	108	113	1		255	255	0
5	89	80	146	134	38		255	255	0
6	56	56	83	110	51		255	255	0
7	34	30	80	145	71		255	255	0
8	57	54	96	124	117		255	255	0
9	87	78	142	134	152		255	255	0
10	94	85	153	134	57		255	255	0
11	255	255	255	83	126		255	255	0
12	0	0	34	150	93		255	128	0
13	69	64	113	127	116		255	128	0
14	70	62	125	139	93		255	128	0
15	52	45	105	146	174		255	128	0

Figura 5.22: Rappresentazione di un quadri mestre con KPI a 7gg

5.2.4 Divisione del dataset

Le finestre temporali create, relative agli anni 2018/2019, saranno utilizzate per il training ed il testing dei vari classificatori, previa opportuna etichettatura. L'insieme di immagini che costituiscono l'anno 2020, invece, riceverà un riscontro sul loro andamento, tramite il miglior modello di classificazione ottenuto nella fase precedente.

Ecco come appaiono alcune immagini create:

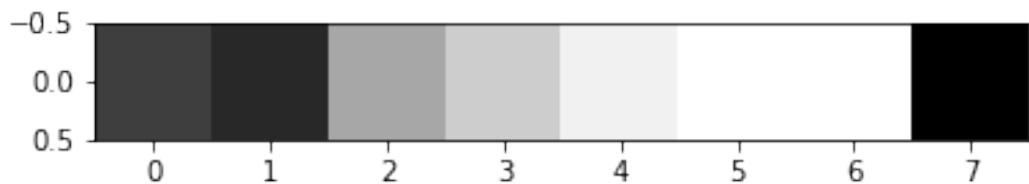


Figura 5.23: Rappresentazione di una settimana

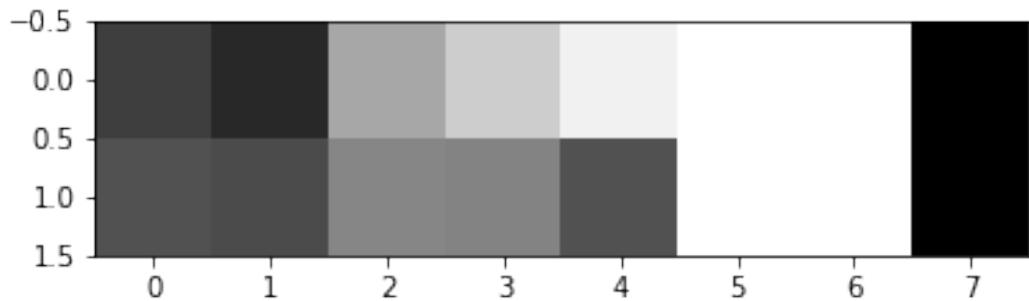


Figura 5.24: Rappresentazione di due settimane con KPI a 7gg

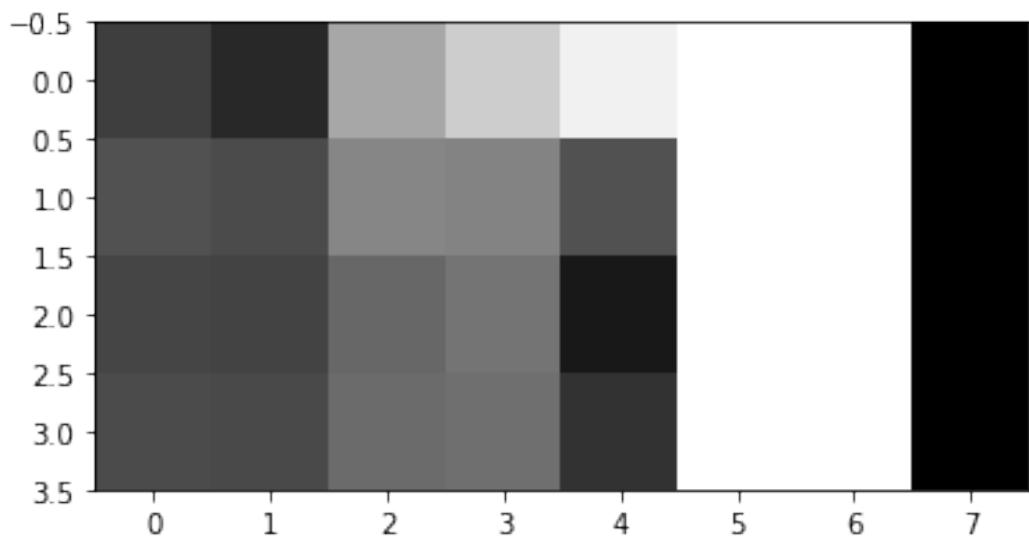


Figura 5.25: Rappresentazione di un mese con KPI a 7gg

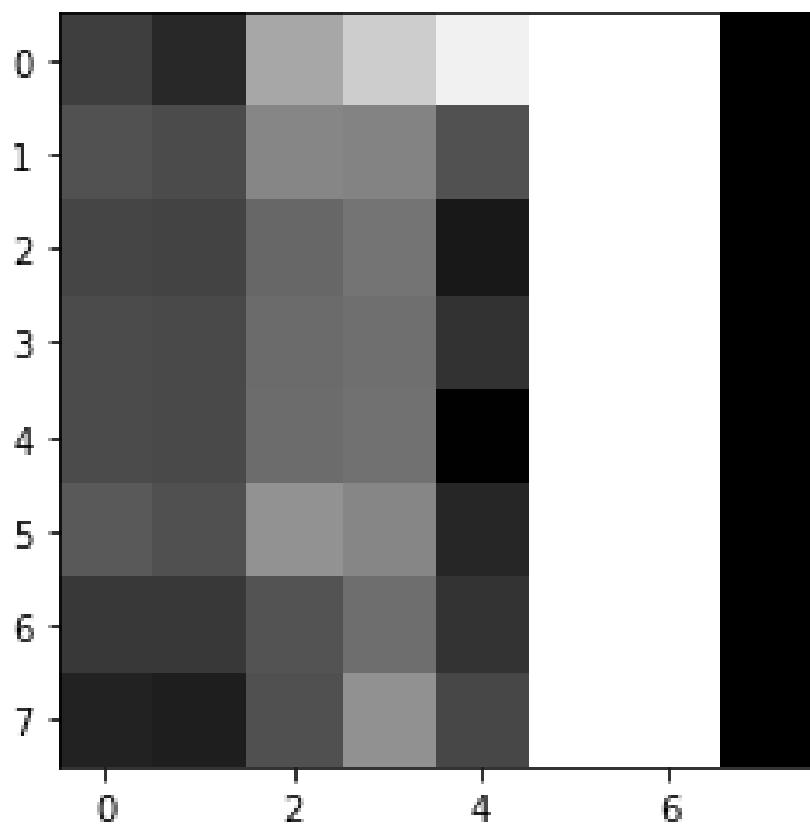


Figura 5.26: Rappresentazione di un bimestre con KPI a 7gg

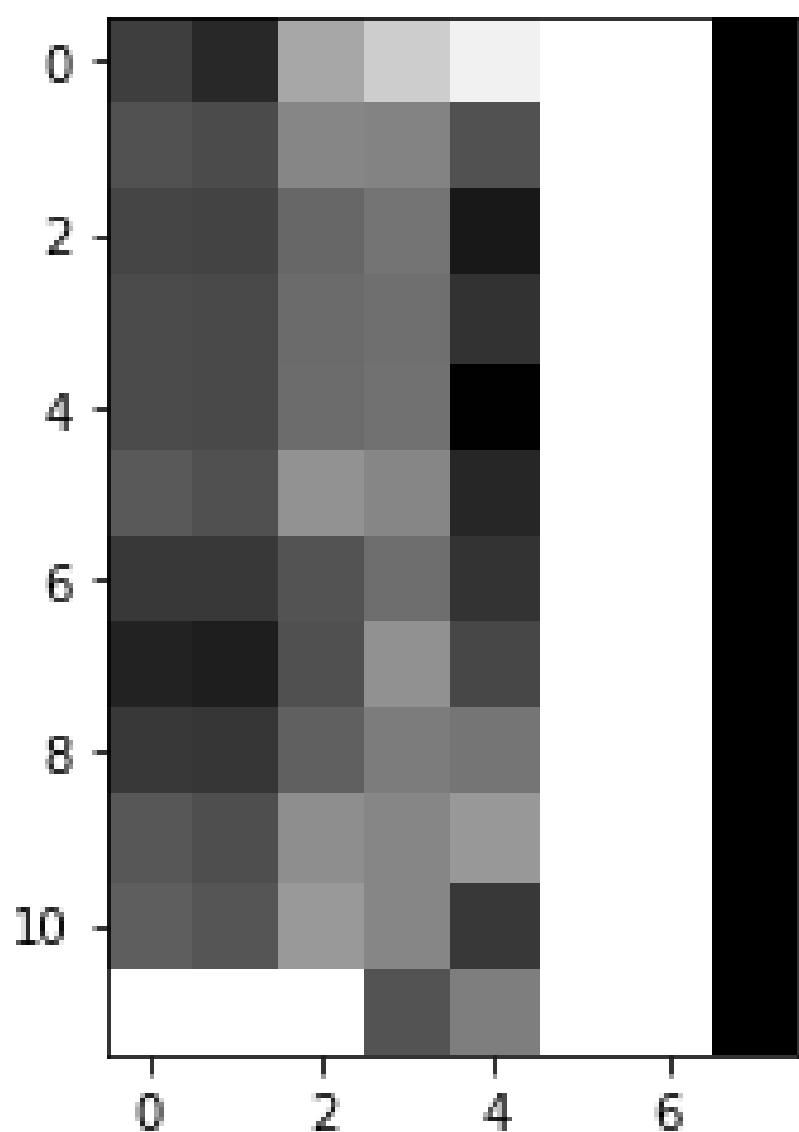


Figura 5.27: Rappresentazione di un trimestre con KPI a 7gg

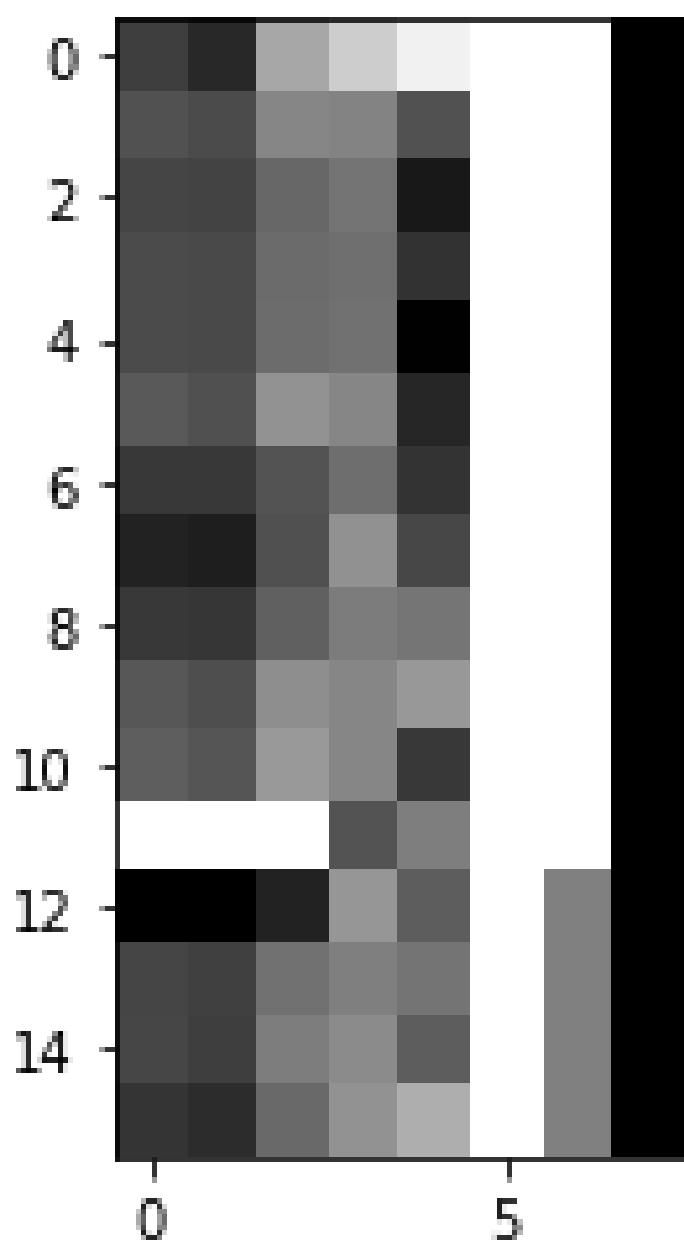


Figura 5.28: Rappresentazione di un quadrimestre con KPI a 7gg

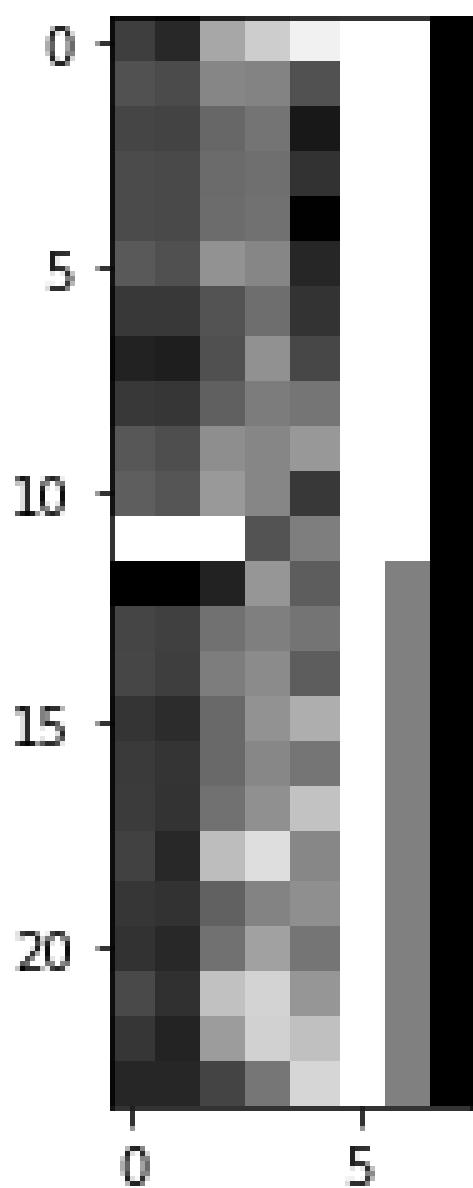


Figura 5.29: Rappresentazione di un semestre con KPI a 7gg

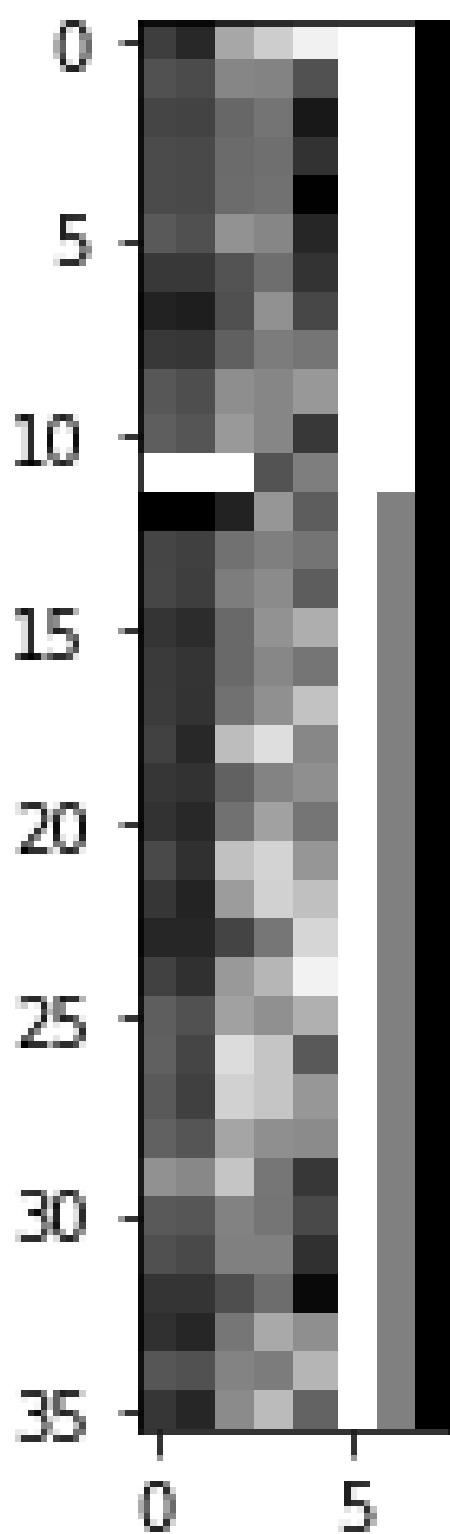


Figura 5.30: Rappresentazione di un nonimestre con KPI a 7gg

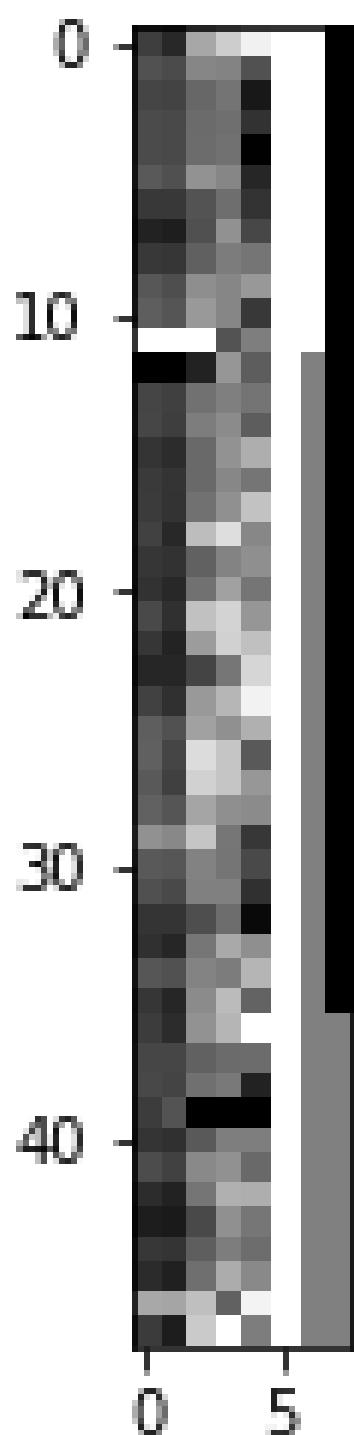


Figura 5.31: Rappresentazione di un anno con KPI a 7gg

La rappresentazione delle finestre con indicatori a 15 o 30 giorni, produce una struttura simile alle precedenti. In particolare, aumentando il periodo considerato dal KPI, si dimezzano le righe per la rappresentazione di una stessa finestra temporale.

Ad esempio, la rappresentazione di un bimestre con KPI a 7gg conterrà 8 righe, a 15gg 4 e a 30gg 2.

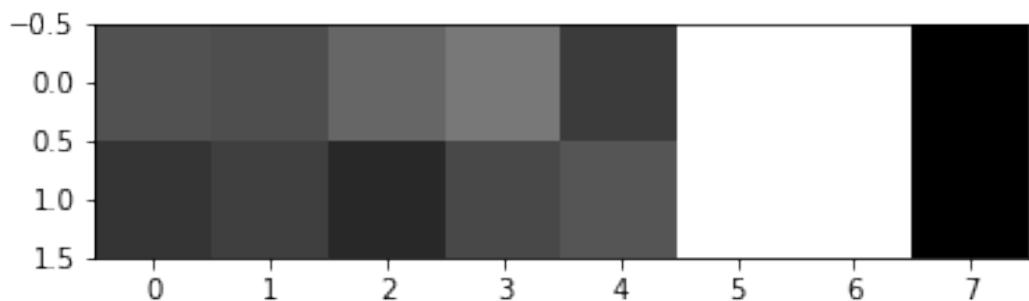


Figura 5.32: Rappresentazione di un bimestre con KPI a 15gg

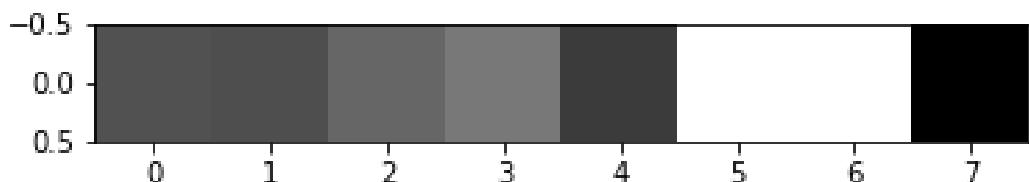


Figura 5.33: Rappresentazione di un bimestre con KPI a 30gg

5.2.5 Etichettatura

Il dataset utilizzato per una prima fase di training e test dei vari classificatori, come specificato nella sezione precedente, deve essere etichettato. L'idea di base è ottenere una scala a 5 livelli:

- **0: Non sufficiente**
- **1: Sufficiente**
- **2: Stabile**
- **3: Positiva**
- **4: Eccellente**

Per la creazione dei vari score, si utilizzerà la combinazione di due funzioni:

- **Funzione 1:** Dato un dataset di immagini D , $\forall img \in D$, si calcola un array $score = \cup_{i=1}^8 KPI_i$, dove ogni singolo punteggio è calcolato a seconda della tipologia del KPI (di valore o di andamento), come da codice sottostante:

```
def generaP1(tabella ,colonna ,righe ):  
    p=0  
    v=tabella [colonna] . values  
    i=1  
    while ( i<len (v )):  
        if (v [ i]>v [ i -1]):  
            p=p+1  
        i=i+1  
return p/righe
```

```
def generaP2( tabella , colonna , righe ):
    p=0
    v=tabella [ colonna ]. values
    for i in v:
        if ( i ==255):
            p=p+1
        elif ( i ==128):
            p=p+0.5
    return p/righe
```

Ad esempio, considerando questo approccio per finestre mensili con KPI a 7gg, otterremo le seguenti tabelle:

	scoreMargineS2018	scoreAverageSpeditoS2018	scoreTrendSpeditoS2018	scoreConsumi2018	scorePrezzi2018
0	0.00	0.25	1.0	1.0	0.0
1	0.50	0.75	1.0	1.0	0.0
2	0.25	0.50	1.0	1.0	0.0
3	0.50	0.50	1.0	0.5	0.0
4	0.50	0.50	1.0	0.5	0.0
5	0.25	0.75	1.0	0.5	0.0
6	0.25	0.25	1.0	0.5	0.0
7	0.25	0.25	1.0	0.5	0.0
8	0.50	0.50	1.0	0.5	0.0
9	0.25	0.00	1.0	0.5	0.5
10	0.50	0.25	1.0	0.5	0.5
11	0.50	0.50	1.0	0.5	0.5

Figura 5.34: Punteggio F1 finestra mensile con KPI a 7gg per l'anno 2018

	scoreMargineS2019	scoreAverageSpeditoS2019	scoreTrendSpeditoS2019	scoreConsumi2019	scorePrezzi2019
0	0.25	0.25	1.00	0.5	0.0
1	0.75	0.50	0.75	0.5	0.0
2	0.25	0.50	1.00	0.5	0.0
3	0.25	0.25	1.00	1.0	0.0
4	0.25	0.25	1.00	1.0	0.0
5	0.25	0.50	1.00	1.0	0.0
6	0.50	0.25	1.00	0.0	0.0
7	0.25	0.50	1.00	0.0	0.0
8	0.50	0.50	1.00	0.0	0.0
9	0.75	0.25	1.00	0.0	0.5
10	0.25	0.25	0.50	0.0	0.5
11	0.25	0.25	1.00	0.0	0.5

Figura 5.35: Punteggio F1 finestra mensile con KPI a 7gg per l'anno 2019

- **Funzione 2:** Si pesano i risultati delle tabelle precedenti, in base ad un target di crescita prefissato (posto realisticamente al 5%), calcolando l'etichetta finale.

In particolare, si farà un confronto dei valori relativi al $KPI_i, \forall i \in [1, 8]$. $\forall riga_j \in tab2018, \forall riga_z \in tab2019$, se $KPI_i(riga_j) > KPI_i(riga_z)$ sarà attribuito un punteggio maggiore all'indicatore i-esimo della riga relativa all'anno 2018, se invece $KPI_i(riga_j) < KPI_i(riga_z)$ ne sarà attribuito uno maggiore all'indicatore i-esimo della riga relativa all'anno 2019, altrimenti se $KPI_i(riga_j) = KPI_i(riga_z)$ entrambi gli indicatori i-esimi riceveranno lo stesso punteggio.

Quindi, $\forall riga_i, riga_j$ la $\sum_{i=1}^8 score(KPI_i)$, rappresenterà l'etichetta relativa alla finestra di visualizzazione in esame. Si precisa che l'upper bound per ogni etichetta è posto a 5, come da obiettivo prefissato.

```

def calcolaEtichette ( tabsP , tabsS , numR):
    labP = []
    labS = []
    for i in range ( numR ):
        lp , ls=calcola (
            tabsP . iloc [ i , : ] . values , tabsS . iloc [ i , : ] . values )
        lp=normal_round ( lp )
        ls=normal_round ( ls )
        labP . append ( lp )
        labS . append ( ls )
    return labP , labS

def calcola ( precedente , successivo ):
    lp=0, ls=0
    for i in range ( len ( precedente ) ):
        lp=lp+precedente [ i ]
        ls=ls+successivo [ i ]
        if ( i >1 ):
            if ( successivo [ i ] > precedente [ i ] ):
                ls=ls +0.5
            elif ( successivo [ i ] < precedente [ i ] ):
                lp=lp +0.5
            elif ( successivo [ i ] == precedente [ i ] ):
                ls=ls +0.25
                lp=lp +0.25
        else :
            if ( successivo [ i ] - precedente [ i ] >= soglie [ i ] ):
                ls=ls +0.5
            elif ( precedente [ i ] - successivo [ i ] >= soglie [ i ] ):
                lp=lp +0.5
    return lp , ls

```

Le tabelle relative all'esempio precedente, produrranno le etichette sottostanti.

```
etichetteMese=calcolaEtichette(s1Mese,s2Mese,12)
etichetteMese

([3, 5, 4, 4, 4, 3, 3, 3, 3, 4, 5], [3, 3, 3, 3, 4, 3, 3, 2, 4, 2, 2])
```

Figura 5.36: Etichette relative alla finestra mensile con KPI a 7gg

L'array di sinistra, rappresenta l'andamento relativo ad ogni mese per l'anno 2018, quello di destra relativo ad ogni mese per l'anno 2019.

Una metodologia analoga, viene applicata ad ogni finestra di KPI creata. Si precisa che le immagini rappresentanti una settimana o due settimane, utilizzano come etichettatura quella prevista per la finestra mensile, contenente le medesime.

Ecco alcune immagini etichettate contenuti nel dataset appena creato:

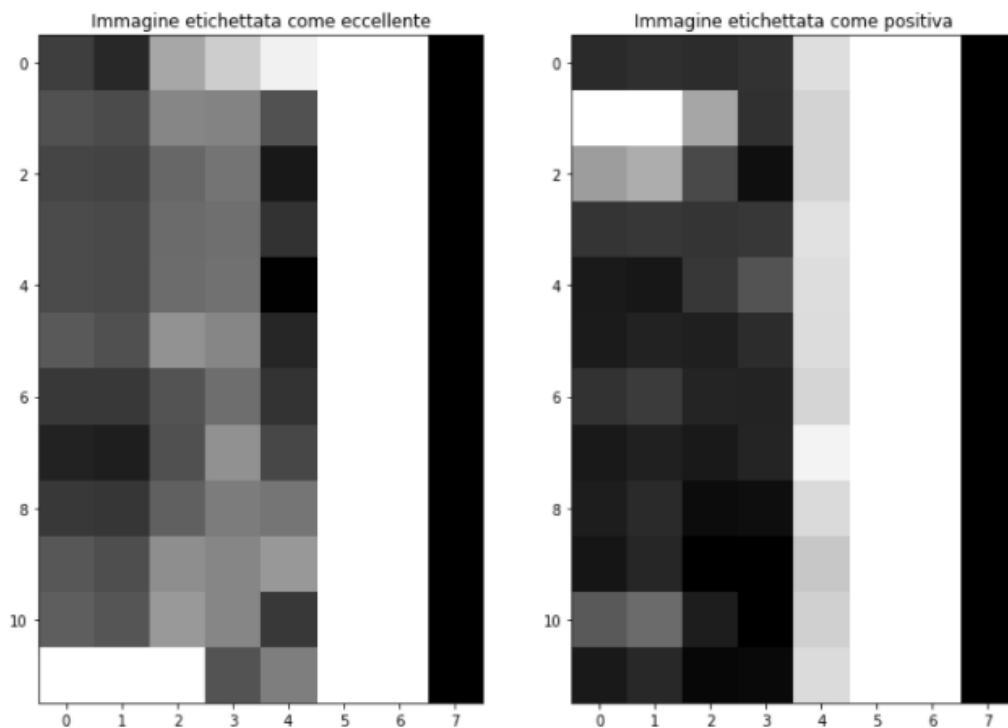


Figura 5.37: Esempio 1 etichettatura

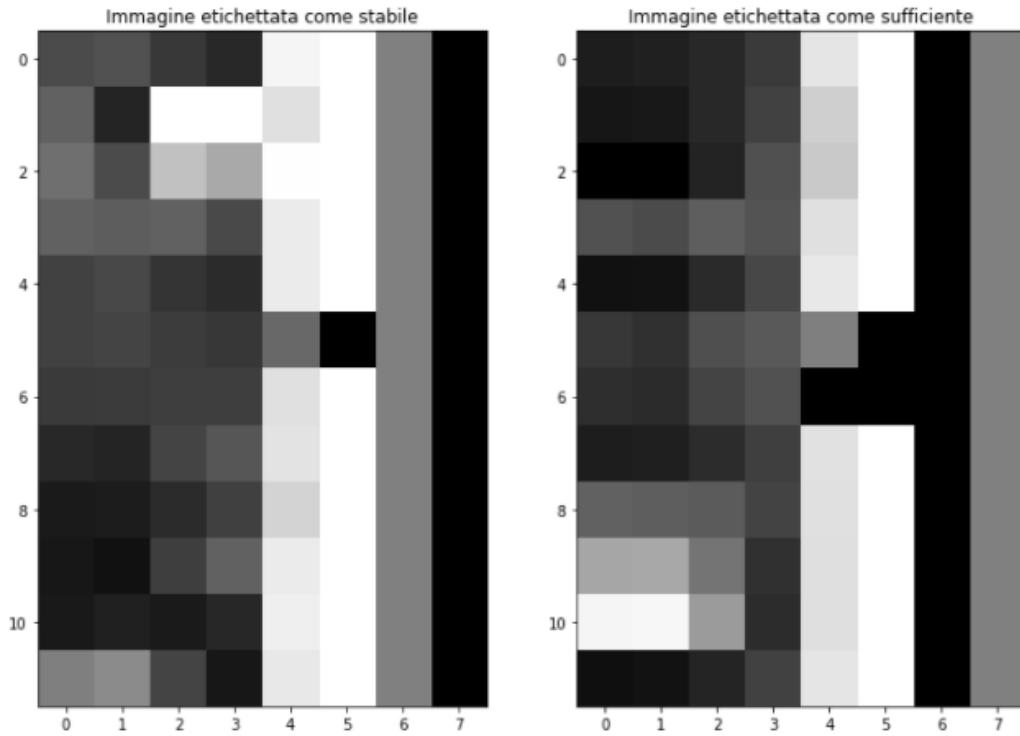


Figura 5.38: Esempio 2 etichettatura

L'etichettatura è coerente con tutta la metodologia utilizzata. Immagini con pixel tendenti a colori chiari rappresentano un migliore andamento rispetto a quelle con alta presenza di pixel scuri.

5.2.6 Preprocessing per immagini e bilanciamento del dataset

Si passa ora ad uniformare le immagini, aggiungendo degli array di zero per tutte quelle con taglia diversa da (48,8). Il dataset appena creato, associando ad ogni immagine la propria etichettatura, subisce operazioni di preprocessing, resize e bilanciamento, tipiche operazioni che garantiranno una migliore analisi da parte dei vari classificatori e della rete implementati.

5.3 Classificazione

In questa sezione, vengono implementati vari modelli di classificazione, dettagliati nel capitolo 5, tramite le librerie Sklearn, Tensorflow e Keras.

Occorre fare una premessa: quando si possiede un quantitativo di dati etichettati, ma limitato, occorre utilizzare un classificatore con distorsione elevata. Al crescere di questa, si avrà una varianza inferiore, il che è positivo quando il dataset è poco numeroso. Se, invece, si hanno a disposizione un sacco di dati, il classificatore non ha molta importanza, ma si dovrebbe semplicemente sceglierne uno con buona scalabilità.

Alla luce di tutto ciò, i classificatori sottostanti utilizzeranno il dataset di dimensione standard, mentre la rete neurale, che ha bisogno di più immagini per la fase di training, riceverà un dataset più ampio, ottenuto banalmente, concatenando più volte il dataset standard.

5.3.1 Tuning dei Classificatori

Questa fase determina, per ogni classificatore, il miglior settaggio possibile. Viene effettuata una prima analisi sul dataset senza l'uso della validazione incrociata, come da codice sottostante.

```
def tuningRandomForest( X_train , y_train ):
    param_rf_grid = { 'n_estimators': [10 ,50 ,100] ,
                      'max_depth': [25 ,50 ,75 ,100] }
    rf = RandomForestClassifier( random_state=42)
    grid_search = GridSearchCV( rf , param_rf_grid , n_jobs = -1)
    grid_search . fit ( X_train , y_train )
    print( " Best_Param:" , grid_search . best_params_ )
    best_rf = grid_search . best_estimator_
    return best_rf
```

```

def tuningAdaBoostClassifier(X_train ,y_train):
    param_ada_grid={ 'n_estimators' :[10 ,50 ,100 ,500] ,
                    'learning_rate' :[0.0001 , 0.001 , 0.01 , 0.1 , 1.0] ,
                    }
    ada=AdaBoostClassifier(
        DecisionTreeClassifier(max_depth=1),
        algorithm='SAMME.R')
    grid_search=GridSearchCV(ada ,param_ada_grid ,
                           scoring='accuracy' ,n_jobs=-1)
    grid_search . fit (X_train , y_train)
    print("Best_Param:" , grid_search . best_params_)
    best_ada = grid_search . best_estimator_
    return best_ada


def tuningKNN(X_train ,y_train ,knn):
    params = { 'n_neighbors' : range(1 , 21 , 2) ,
               'weights' :[ 'uniform' , 'distance' ] ,
               'metric' :[ 'euclidean' , 'manhattan' , 'minkowski' ]
               }
    grid_search = GridSearchCV(knn , params ,
                               scoring='accuracy' ,n_jobs=-1)
    grid_search . fit (X_train , y_train)
    print("Best_Param:" , grid_search . best_params_)
    best_KNN = grid_search . best_estimator_
    return best_KNN


def tuningSVC(X_train ,y_train ,svc):
    param_svc_grid ={ 'kernel' : [ 'poly' , 'linear' , 'rbf' , 'sigmoid' ] ,
                      'C' : [ 500 , 300 , 200 , 150 , 100 , 50 , 10 , 1] ,
                      'gamma' : [ 'scale' ]
    }

```

```
        }
```

```
grid_search = GridSearchCV(svc, param_svc_grid,
                           scoring='accuracy', n_jobs=-1)
grid_search.fit(X_train, y_train)
print("Best_Param:", grid_search.best_params_)
best_svc = grid_search.best_estimator_
return best_svc
```



```
class MyClassifierTrain(BaseEstimator, TransformerMixin):
```

```
    def __init__(self):
        pass
```

```
    def fit(self, X, y):
        return self
```

```
    def transform(self, X,y):
        X=X.reshape(len(X),100*100)
        X_train , X_test , y_train , y_test=train_test_split
                           (X, y ,
                           test_size=0.2,
                           shuffle=True)
```

```
        print("Training_modelli...")
```

```
        print("AdaBoost...")
```

```
        ada_best=tuningAdaBoostClassifier(X_train , y_train)
```

```
        print("Tuning_SVC...")
```

```

svc_clf=SVC()
svc_best = tuningSVC(X_train ,y_train ,svc_clf)
print("Tuning_KNN... ")
knn_clf=KNeighborsClassifier()
knn_best=tuningKNN(X_train ,y_train ,knn_clf)
print("Random_Forest ... ")
rf_best=tuningRandomForest(X_train ,y_train )
print("Fatto!")

```

L'algoritmo restituisce la configurazione da adottare, per ogni classificatore da utilizzare, nella fase di training.

```

Training modelli...
AdaBoost...
Best Param: {'learning_rate': 0.001, 'n_estimators': 100}
Tuning SVC...
Best Param: {'C': 300, 'gamma': 'scale', 'kernel': 'rbf'}
Tuning KNN...
Best Param: {'metric': 'euclidean', 'n_neighbors': 1, 'weights': 'uniform'}
Random Forest...
Best Param: {'max_depth': 25, 'n_estimators': 100}
Fatto!

```

Figura 5.39: Tuning classificatori

5.3.2 Tuning LeNet-5

Per quanto concerne la rete neurale scelta, si è definita un'implementazione di LeNet-5 "modernizzata", tramite il pacchetto Keras, sostituendo i filtri di average pooling con quelli di max pooling. Inoltre, passare ad immagini con risoluzione più elevata di quella standard permette, mantendendo invariate le dimensioni dei filtri, di effettuare una sorta di crop naturale di ogni elemento; in grado di cogliere in maniera più sottile le caratteristiche peculiari di ogni

immagine, con una conseguente miglioria delle performance.

Tramite lo studio della funzione di loss e dell'accuratezza, si cerca di capire il numero di epoche necessarie alla fase di training, ai fini di ottenere buoni risultati ed evitare fenomeni di overfitting della rete.

```
def create_network():
    lenet = Sequential()
    lenet.add(Conv2D(filters=32, kernel_size=(5,5), padding='same',
                     activation='relu',
                     input_shape=(100, 100, 1)))
    lenet.add(MaxPool2D(strides=2))
    lenet.add(Conv2D(filters=48, kernel_size=(5,5),
                     padding='valid', activation='relu'))
    lenet.add(MaxPool2D(strides=2))
    lenet.add(Flatten())
    lenet.add(Dense(256, activation='relu'))
    lenet.add(Dense(84, activation='relu'))
    lenet.add(Dense(4, activation='softmax'))
    adam = Adam(learning_rate=5e-4)
    lenet.compile(loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'], optimizer=adam)
    return lenet
```

```
history = model.fit(X_train, y_train,
                     batch_size=100,
                     steps_per_epoch=len(X_train) / 100,
                     epochs=epochs,
                     validation_data=(X_test, y_test),
                     callbacks=[reduce_lr], verbose=0)

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model_loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper_left')
plt.show()

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model_accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper_left')
plt.show()

class MyClassifierTrainNN(BaseEstimator, TransformerMixin):

    def __init__(self):
        pass

    def fit(self, X, y):
        return self
```

```
def transform(self, X, y, epochs):
    print("Preprocessing per NN...")
    X, y = preNN(X, y, i, j)
    X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                       test_size=0.2)

    print("Tuning NN...")
    tuningNN(X_train, X_test, y_train, y_test, epochs)

MyClassifierTrainNN().transform(X, y, 60)
```

Otterremo i seguenti risultati:

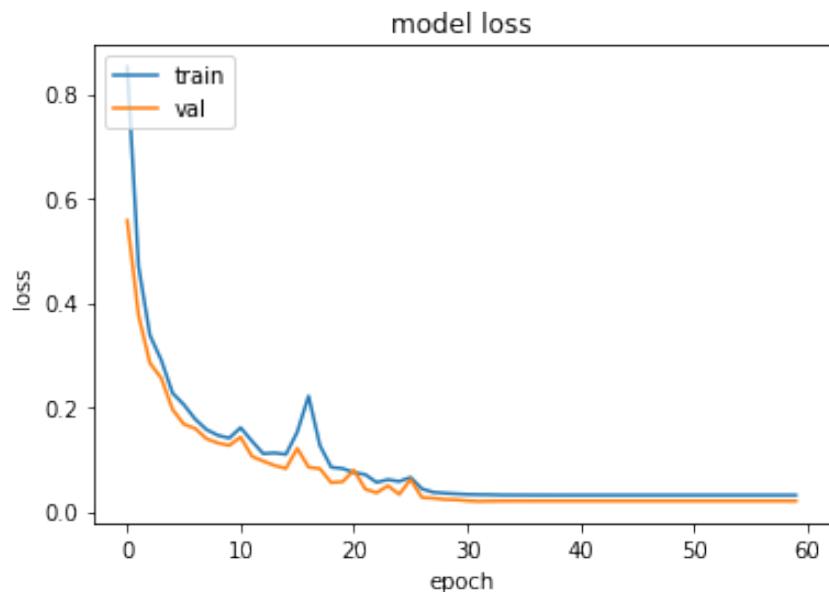


Figura 5.40: Loss LeNet-5 su 60 epoche

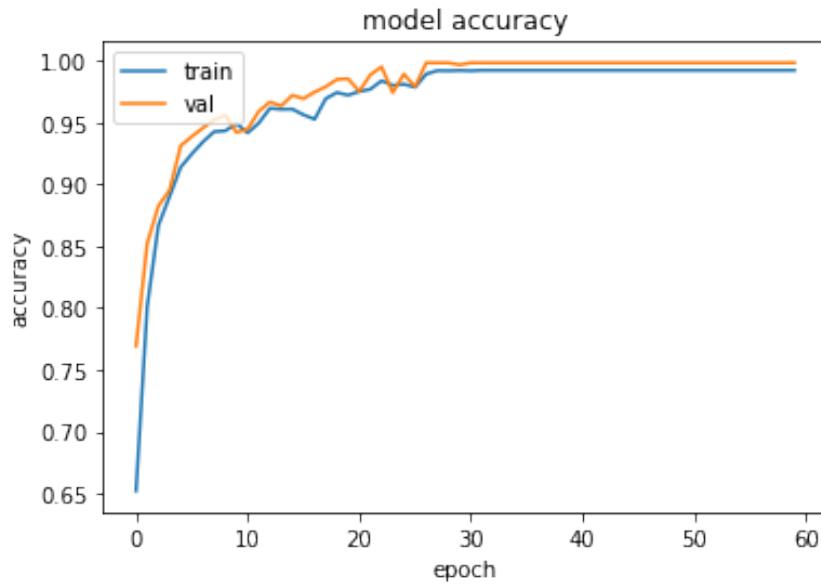


Figura 5.41: Accuratezza LeNet-5 su 60 epoche

Risulta evidente che il numero di epoche necessarie è 35, dopo le quali il modello si stabilizza.

5.3.3 Fase di training

Per la fase di training, si utilizzerà la convalida incrociata a 10 sacche.

```
kf=KFold( n_splits=10)
score_KFold = []
```

Tutte le metodologie sopra elencate, effettuano questa convalida sul dataset in esame, restituendo sia l'accuratezza media, sia la deviazione std ottenuta.

In particolare viene salvato, per ogni metodo di classificazione, il modello migliore generato sulle 10 possibili sacche; mostrando i risultati ottenuti, in termini di matrice di confusione, score di precision, recall e f1, inerenti a questo insieme.

5.3.4 Analisi dei risultati

Come si può ammirare dai grafici sottostanti, AdaBoost non si presta molto bene all'analisi di un problema di questo tipo e può già essere scartato.

```
L'accuratezza per ogni sacca della cross validation è rappresentata dal seguente vettore
[0.69230769 0.6025641 0.51282051 0.57692308 0.66666667 0.58974359
 0.5974026 0.62337662 0.64935065 0.63636364]
Ottenendo un'accuratezza media di 0.615 con deviazione standard di 0.048
La migliore accuratezza e' 0.692 ottenuta nello split numero 1
Salvo il modello ottenuto nello split numero 1
Si mostra ora la matrice di adiacenza ottenuta sul test set dello split numero 1
```

Figura 5.42: Risultati AdaBoost

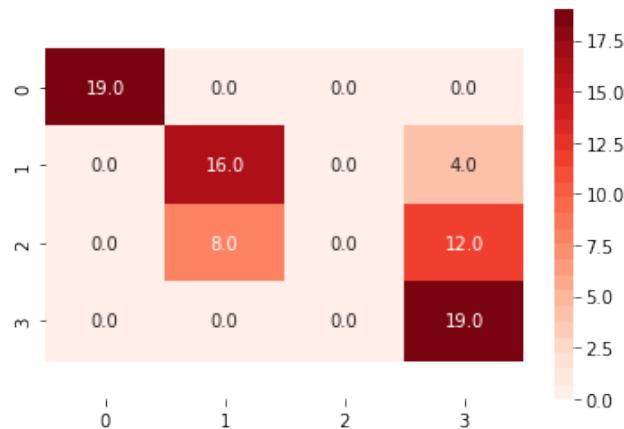


Figura 5.43: Matrice di confusione AdaBoost

	precision	recall	f1-score	support
sufficiente	1.00	1.00	1.00	19
stabile	0.67	0.80	0.73	20
positivo	0.00	0.00	0.00	20
eccellente	0.54	1.00	0.70	19
accuracy			0.69	78
macro avg	0.55	0.70	0.61	78
weighted avg	0.55	0.69	0.60	78

Figura 5.44: Score sacca migliore Adaboost

L'SVC mostra buoni risultati, come da figure sottostanti, comportandosi meglio del precedente classificatore. Tuttavia non garantisce, in questo contesto, una performance che può essere ritenuta molto soddisfacente.

```
L'accuratezza per ogni sacca della cross validation è rappresentata dal seguente vettore
[0.73076923 0.78205128 0.76923077 0.83333333 0.88461538 0.82051282
 0.90909091 0.87012987 0.87012987 0.84415584]
Ottenendo un'accuratezza media di 0.831 con deviazione standard di 0.053
La migliore accuratezza e' 0.909 ottenuta nello split numero 7
Salvo il modello ottenuto nello split numero 7
Si mostra ora la matrice di adiacenza ottenuta sul test set dello split numero 7
```

Figura 5.45: Risultati SVC



Figura 5.46: Matrice di confusione sacca migliore SVC

	precision	recall	f1-score	support
sufficiente	0.90	1.00	0.95	19
stabile	0.94	0.79	0.86	19
positivo	0.85	0.89	0.87	19
eccellente	0.95	0.95	0.95	20
accuracy			0.91	77
macro avg	0.91	0.91	0.91	77
weighted avg	0.91	0.91	0.91	77

Figura 5.47: Score sacca migliore SVC

I primi risultati, con ottima garanzia affidabilità, sono prodotti dal classificatore KNN, arrivando ad un'accuratezza media dell' 86%. Nella sacca migliore, raggiunge addirittura il 96% di accuratezza, mostrando buoni valori nella matrice di confusione, con buone performance in precision, recall ed f1-score.

```
L'accuratezza per ogni sacca della cross validation è rappresentata dal seguente vettore
[0.8974359  0.85897436 0.87179487 0.79487179 0.82051282 0.88461538
 0.96103896 0.8961039  0.87012987 0.79220779]
Ottenendo un'accuratezza media di 0.865 con deviazione standard di 0.049
La migliore accuratezza e' 0.961 ottenuta nello split numero 7
Salvo il modello ottenuto nello split numero 7
Si mostra ora la matrice di adiacenza ottenuta sul test set dello split numero 7
```

Figura 5.48: Risultati KNN



Figura 5.49: Matrice di confusione sacca migliore KNN

	precision	recall	f1-score	support
sufficiente	0.95	1.00	0.97	19
stabile	1.00	0.84	0.91	19
positivo	0.90	1.00	0.95	19
eccellente	1.00	1.00	1.00	20
accuracy			0.96	77
macro avg	0.96	0.96	0.96	77
weighted avg	0.96	0.96	0.96	77

Figura 5.50: Score sacca migliore KNN

Sebbene non sia necessario utilizzare un cross validation su un classificatore Random Forest, in virtù dei meccanismi dell'out of bag, si procede ugualmente, ai fini di effettuare un confronto paritario con gli altri classificatori in esame. Il risultato è eccellente, avendo un'accuratezza media di circa il 91%, mostrando nella migliore sacca, uno score di accuratezza del 97%, con eccellenti valori visualizzati nella matrice di confusione, sia nelle metriche di precision, recall ed f1-score.

```
L'accuratezza per ogni sacca della cross validation è rappresentata dal seguente vettore
[0.88461538 0.93589744 0.87179487 0.92307692 0.92307692 0.88461538
 0.97402597 0.92207792 0.94805195 0.81818182]
Ottenendo un'accuratezza media di 0.909 con deviazione standard di 0.042
La migliore accuratezza e' 0.974 ottenuta nello split numero 7
Salvo il modello ottenuto nello split numero 7
Si mostra ora la matrice di adiacenza ottenuta sul test set dello split numero 7
```

Figura 5.51: Risultati Random Forest



Figura 5.52: Matrice di confusione sacca migliore Random Forest

	precision	recall	f1-score	support
sufficiente	1.00	1.00	1.00	19
stabile	1.00	0.89	0.94	19
positivo	0.95	1.00	0.97	19
eccellente	0.95	1.00	0.98	20
accuracy			0.97	77
macro avg	0.98	0.97	0.97	77
weighted avg	0.98	0.97	0.97	77

Figura 5.53: Score sacca migliore Random Forest

Infine, si analizzano i risultati della rete neurale convoluzionale adottata. Come volevasi dimostrare, ottiene le migliori performance in termini di accuratezza media, con il 99% di precisione. Nella sacca migliore, si ottiene uno score di quasi il 100%. Ovviamente, sia in termini di matrice di confusione che di precision, recall ed f1-score, si ottengono le performance più alte registrate sulla sacca migliore.

Si precisa che, in virtù delle premesse iniziale, la rete neurale ha bisogno di più esempi di apprendimento, rispetto ai tradizionali classificatori, come mostrato anche dal numero di esempi maggiore contenuti nella matrice di confusione.

L'accuratezza per ogni sacca della cross validation è rappresentata dal seguente vettore
[0.9774678349494934, 0.9828326106071472, 0.9914070963859558, 0.9914070963859558, 0.9903329610824585,
0.980665922164917, 0.9914070963859558, 0.989258885383606, 0.9967776536941528, 0.994629442691803]
Ottenendo un'accuratezza media di 0.989 con deviazione standard di 0.006
La migliore accuratezza e' 0.997 ottenuta nello split numero 9
Salvo il modello ottenuto nello split numero 9
Si mostra ora la matrice di adiacenza ottenuta sul test set dello split numero 9

Figura 5.54: Risultati LeNet-5



Figura 5.55: Matrice di confusione sacca migliore LeNet-5

	precision	recall	f1-score	support
sufficiente	0.98	0.97	0.98	233
stabile	1.00	0.98	0.99	232
positivo	0.97	1.00	0.99	233
eccellente	1.00	1.00	1.00	233
accuracy			0.99	931
macro avg	0.99	0.99	0.99	931
weighted avg	0.99	0.99	0.99	931

Figura 5.56: Score sacca migliore LeNet-5

Tutta questa fase di analisi, è sintetizzata nel grafico sottostante.

Si noti, inoltre, come tutte le premesse fatte in termini di distorsione, sono riflesse nella sperimentazione, risultando veritieri. Il raggio di ogni circoferenza, infatti, è rappresentato dalla distorsione ottenuta dal modello rappresentato.

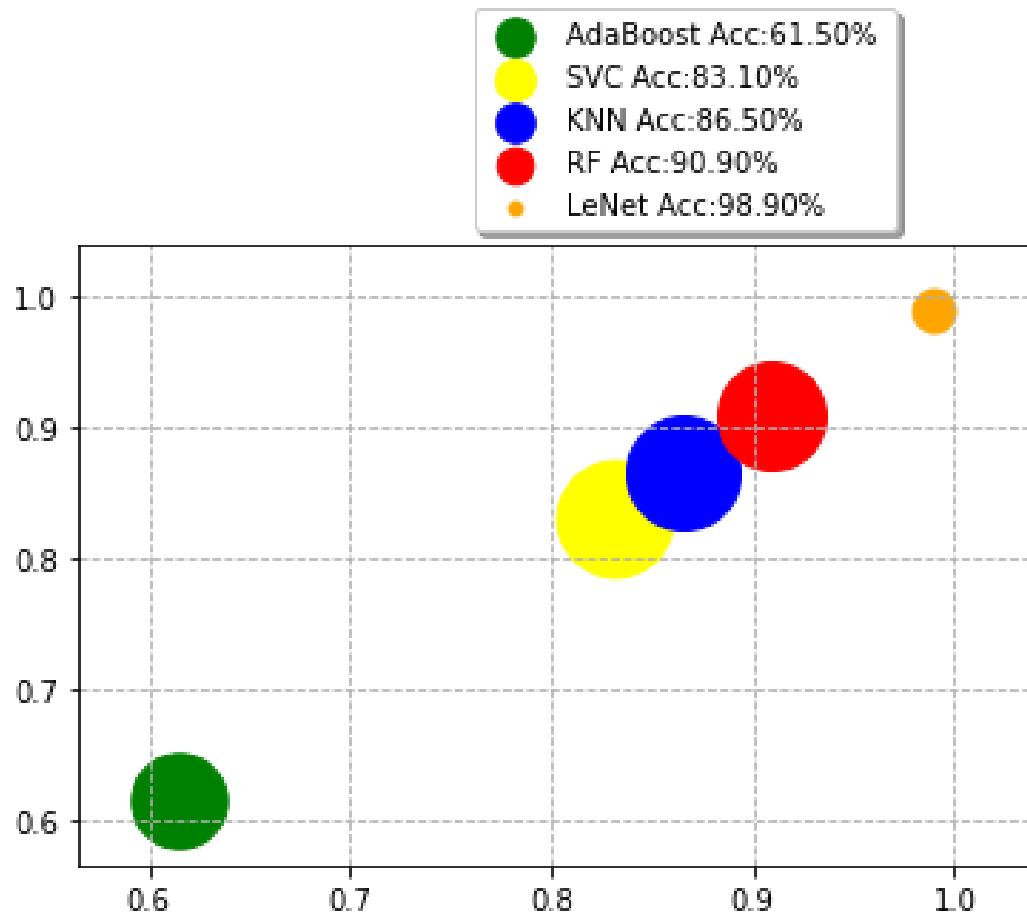


Figura 5.57: Grafico riassuntivo 10 fold

5.3.5 Previsioni sull'anno 2020

Il modello della CNN addestrato, sarà utilizzato per la predizione sull'etichetta dell'anno 2020, mai visto in precedenza. Si mostra, a titolo di esempio, la predizione su finestre di visualizzazioni più frequenti per report aziendali (trimestre, semestre, anno), rispecchiando comunque un buon andamento anche per tutte le altre tipologie e annate considerate. Si parte dal trimestre, scelto con KPI a 15 giorni, che sarà confrontato con i dati nella stessa finestra temporale dell'anno precedente. Il primo trimestre è etichettato come positivo, al contrario dell'anno 2019 etichettato come stabile. La classificazione è corretta, poichè l'immagine di sinistra ha un alta presenza di indicatori tendenti al bianco, al contrario quella di destra contenente molti indicatori peggiori, di colore nero.

Il secondo trimestre, invece, è etichettato come positivo. Infatti, mantiene un andamento simile a quello del 2019, etichettato in precedenza anch'esso come positivo.

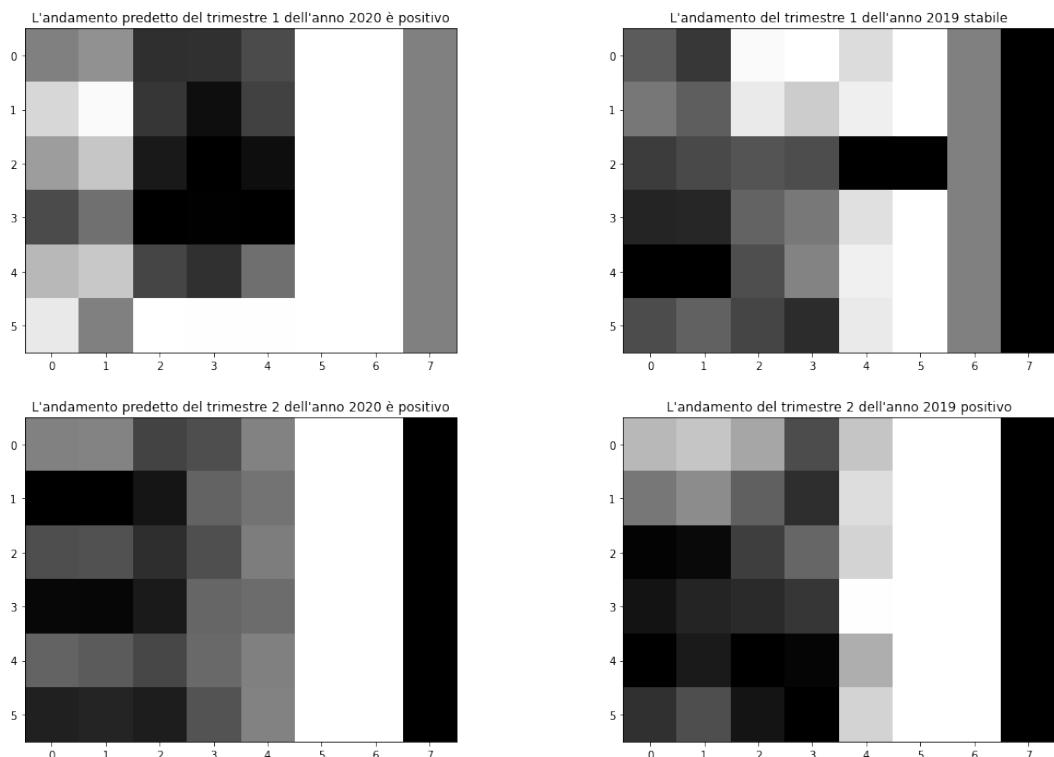


Figura 5.58: Predizione trimestre 1 e 2 anno 2020 con KPI a 15 gg

I trimestri 3 e 4 si comportano meglio rispetto ai corrispondenti del 2019, cosa facilmente intuibile dalla visualizzazione delle finestre di rappresentazione.

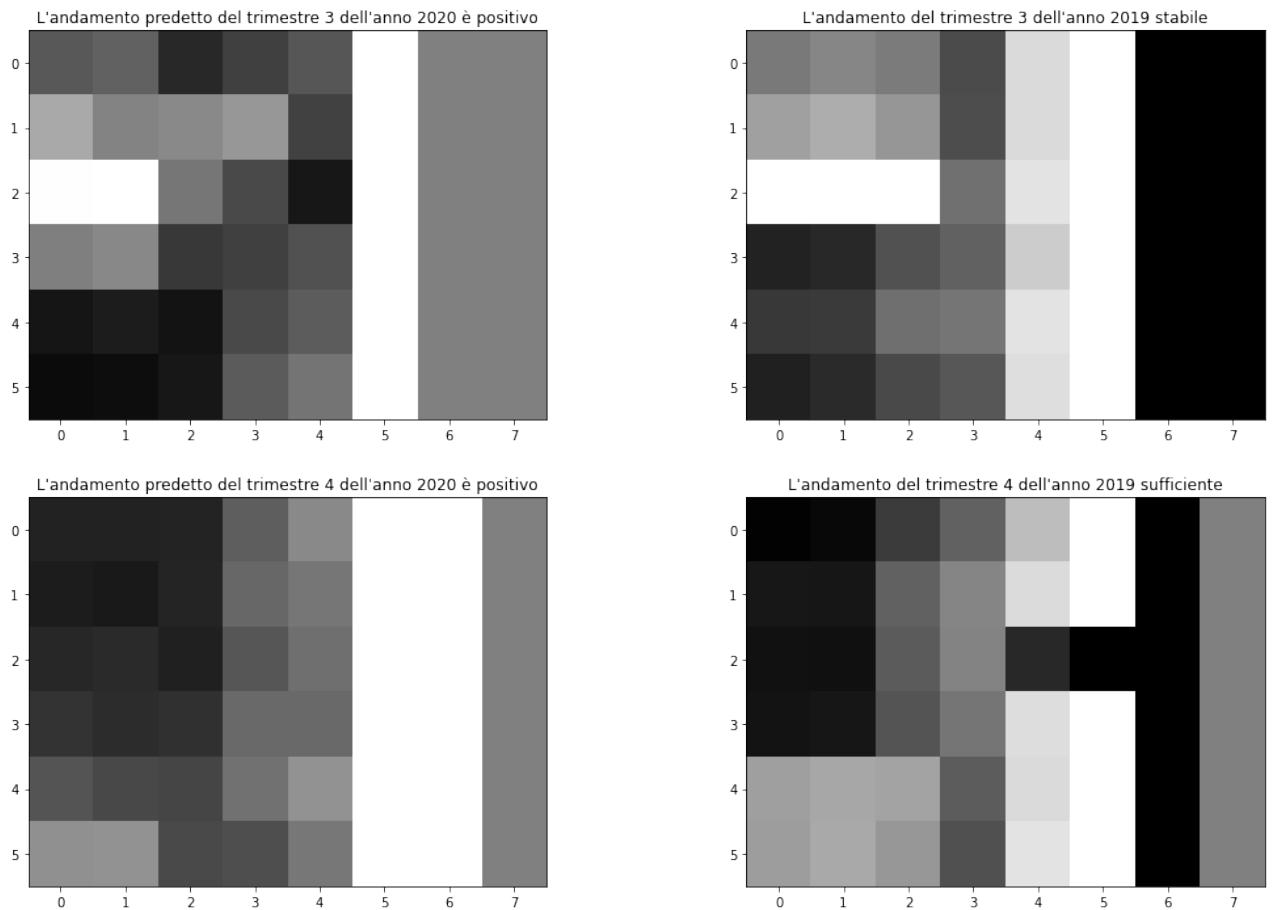


Figura 5.59: Predizione trimestre 3 e 4 anno 2020 con KPI a 15 gg

Utilizzando una finestra con KPI a più largo raggio, 30gg, si evince l'andamento migliore dell'anno 2020 rispetto a quello 2019.

Visualizzando, ad esempio, i risultati sull'andamento semestrale, sono palesi le performance migliori ottenute, presentando molti più valori tendenti al bianco rispetto all'anno 2019.

I semestri del 2020, saranno etichettati come positivi, al contrario di quelli 2019 considerati, correttamente, stabili dalla funzione di etichettatura.

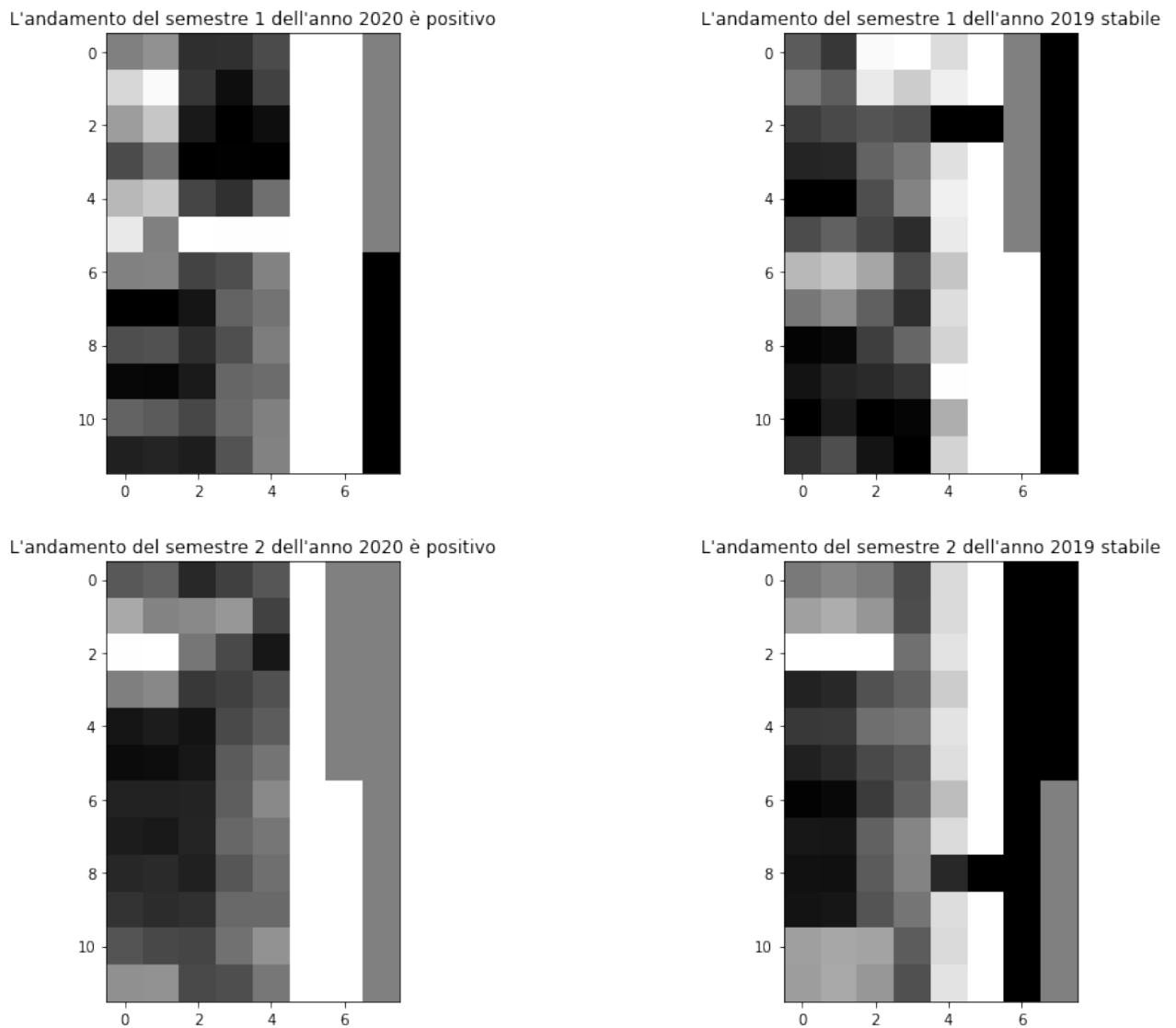


Figura 5.60: Predizione semestre 1 e 2 anno 2020 con KPI a 30 gg

La visualizzazione della finestra temporale di un anno, con KPI a 30gg, rimarca il discorso precedente.

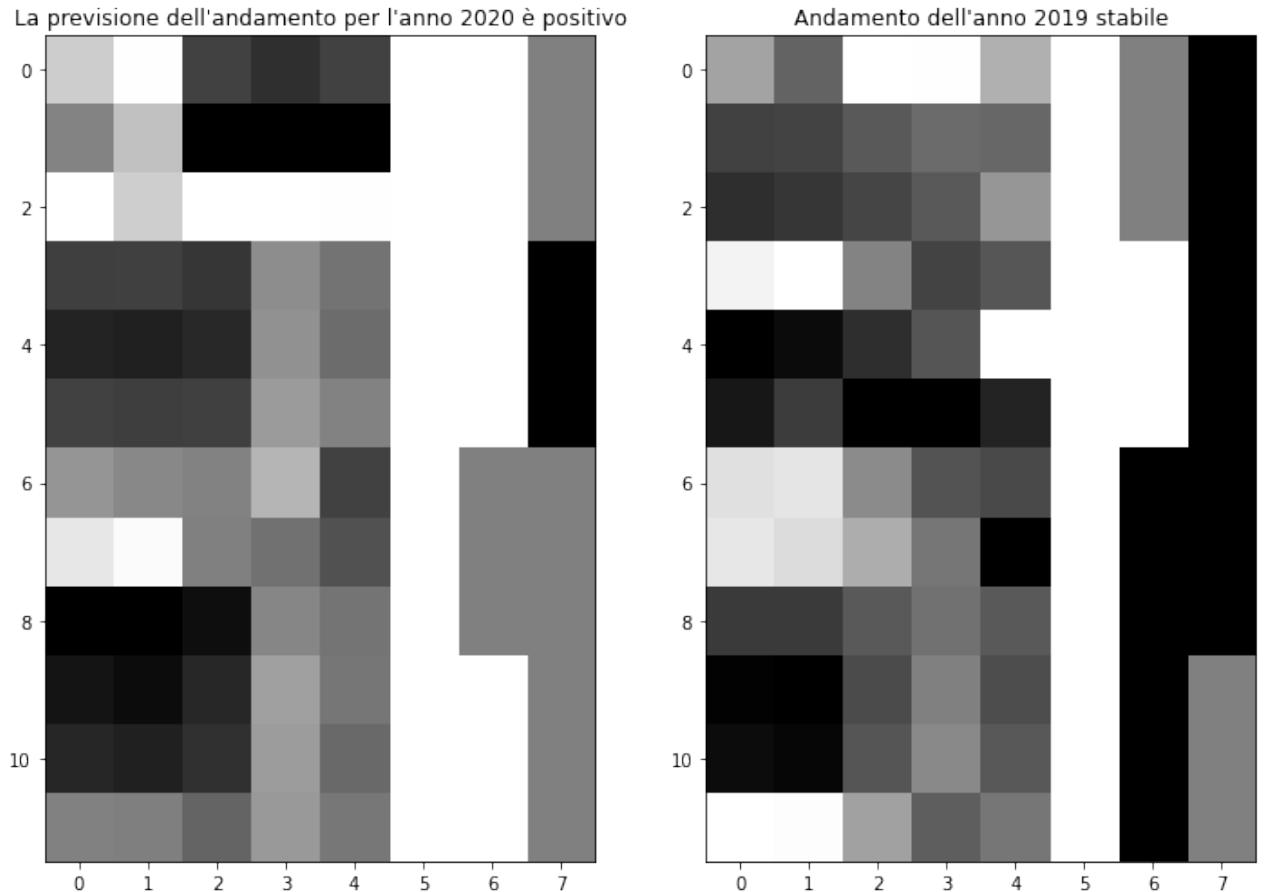


Figura 5.61: Predizione anno 2020 con KPI a 30 gg

Il modello si comporta molto bene in fase di predizione, come da esempi precedenti e come palesato dall'applicazione della stessa predizione a finestre di visualizzazione diversa, rimarcando l'ottimalità della metodologia sperimentale ideata. Infatti, i dati rispecchiano l'esatto responso ottenuto con i classici metodi di BI. Sebbene nel 2020 si è dovuto far fronte alla pandemia da Covid-19, come rispecchiato dai report Ismea[7], il settore alimentare ha registrato un andamento annuo migliore del 2019 di circa un 7%. Il tutto è stato causato da un consumo più eccessivo di prodotti, inerenti al settore agroalimentare, nei mesi in cui si è stati di più in casa, con una conseguente migliorata delle performance aziendali del cliente a cui fa riferimento il dataset utilizzato.

5.4 Anomaly Detection

Si mostra, ora, come la trasformazione del dataset in immagini, si presta molto bene al rilevamento di anomalie.

Dall'unione del triennio di dati in analisi, otteniamo la seguente composizione:

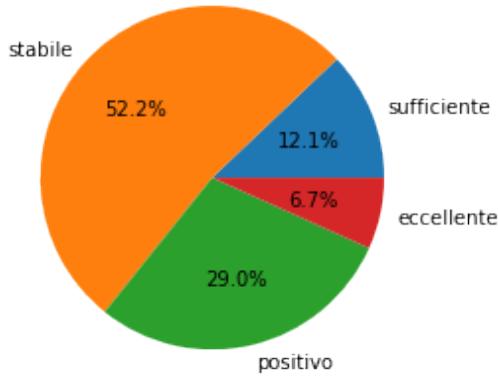


Figura 5.62: Distribuzione degli esempi

Utilizzando un classificatore PCA, si mostra il comportamento del modello nel rilevare tutte le etichette diverse da quelle stabili identificandole, per mostrare la bontà della metodologia adottata, come anomalie.

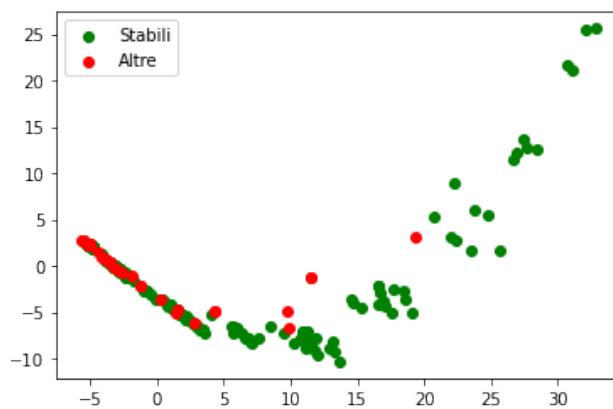


Figura 5.63: Distribuzione delle anomalie (in rosso)

L’analisi delle componenti principali, è una metodologia che rientra nei problemi di trasformazione lineare, ampiamente utilizzata in diversi contesti applicativi, soprattutto per l’estrazione delle caratteristiche e la riduzione della dimensionalità. Le caratteristiche di un determinato set di dati possono essere molte, a volte decisamente troppe. Quello che in data science bisogna evitare, è di trovarsi a trattare con talmente tante dimensioni da non riuscire più a far girare il modello previsionale richiesto. PCA, permette di trovare le direzioni della massima varianza nei dati ad alta dimensionalità e di proiettarle su un nuovo sottospazio con dimensioni uguali o inferiori a quello originale. Per il rilevamento delle anomalie, PCA analizza ogni nuovo input. L’algoritmo, calcola la proiezione sugli eigenvector, insieme a un errore di ricostruzione normalizzato. L’errore viene usato come punteggio delle anomalie. A un punteggio maggiore, corrisponde una maggiore anomalia dell’istanza.

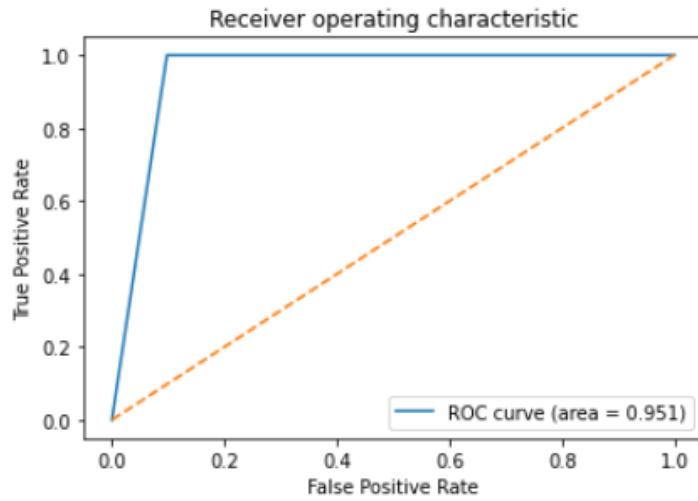


Figura 5.64: Distribuzione delle anomalie

Tramite l’utilizzo di una PCA a 42 componenti, settaggio ottenuto dopo una fase di tuning, si visualizza una curva di roc eccellente, con auc score pari a 0.95. Si mostra, dunque, un comportamento ottimale del modello prodotto su un dataset di questo tipo, rimarcando ulteriormente la bontà della metodologia adottata per la fase di sperimentazione dell’elaborato.

Capitolo 6

Conclusioni e scenari di sviluppo futuri

Il seguente elaborato ha mostrato come la visualizzazione tramite immagini, ottenute attraverso i KPI implementati nel sistema EVO-BI, insieme all'utilizzo di tecniche ben note di machine learning e deep learning, dotate di algoritmi ben rodati e performanti, permette di costruire modelli in grado effettuare analisi e previsioni aziendali.

Dopo aver effettuato le opportune operazioni sul dataset creato ed aver applicato una funzione di etichettatura per gli anni 2018 e 2019, si passa alla fase di classificazione dell'anno 2020. La previsione è stata effettuata dalla rete LeNet-5 implementata, modello migliore in base alle metriche adottate, come si ci poteva aspettare dal largo utilizzo di architetture CNN nell'image recognition.

Inoltre, si è dimostrato come la trasformazione dei dati in immagini di questo tipo, consente facilmente di rilevare eventuali anomalie, tramite ad esempio, l'algoritmo PCA.

In futuro, ottenendo più dati storici su cui allenare e testare il modello, magari contenenti tipologie di clienti appartenenti a diverse aree di mercato, si potrebbe arriverebbe facilmente alla realizzazione di un modello multiarea. Inoltre, avere più dati a disposizione, consentirebbe di utilizzare un approccio implementativo basato su tensori. Infatti, una struttura di questo tipo, potrebbe

contenere al suo interno, una medesima finestra di visualizzazione, con KPI temporali a 7,15 e 30 giorni, mostrano il tutto in un'unica immagine a colori. D'altro canto, l'utilizzo di tensori, richiederà un costo più gravoso, causato dall'utilizzo di reti neurali più complesse.

Ringraziamenti

Un ringraziamento speciale va ai miei genitori Anna e Francesco, a mio fratello Federico, alla mia fidanzata Elisa che, insieme a Dario,Maria Pia e Maria, mi hanno sempre sostenuto, spronandomi sempre a dare il meglio anche in momenti difficili.

Ci tengo a ringraziare la prof.Guzzo, il prof.Scarcello e tutto il team di EVO-BI, per l'opportunità di collaborazione che mi hanno concesso, permettendomi di accrescere il mio bagaglio professionale.

Un ringraziamento speciale va anche al prof.Legato, per il suo supporto ed suoi consigli.

Ringrazio anche "Buccio" e "Buccia" per le nostre lunghe chiacchierate sul mondo Tech, che tanto danno noia ad Elisa, insieme a tutti gli amici, vicini e lontani, che ritengono di aver contribuito alla stesura del presente elaborato. Infine, un elogio speciale a me stesso poichè, se ancora ce ne fosse il bisogno, ho dimostrato di poter superare qualsiasi ostacolo che la vita mi pone dinanzi, raggiungendo i miei obiettivi.

Bibliografia

- [1] <https://ichi.pro/it/comprendione-della-matrice-di-confusione-richiamo-di-precisione-e-punteggio-f1-141157474959377>.
- [2] <https://it.emcelettronica.com/architetture-di-reti-neurali-per-il-deep-learning>.
- [3] <https://spremutedigitali.com/machine-learning-deep-learning-reti-neurali-differenza/>.
- [4] <https://www.ai4business.it/intelligenza-artificiale/deep-learning/reti-neurali/>.
- [5] <https://www.datawiring.me/tipi-di-algoritmi-per-il-machine-learning/>.
- [6] <https://www.internet4things.it/iot-library/image-recognition-cose-come-funziona-e-i-vantaggi-per-le-aziende/>.
- [7] <https://www.ismea.it/istituto-di-servizi-per-il-mercato-agricolo-alimentare>.
- [8] <https://www.lorenzogovoni.com/5-tecniche-di-cross-validation/>.
- [9] <https://www.lorenzogovoni.com/architettura-di-rete-neurale-convoluzionale/>.
- [10] <https://www.lorenzogovoni.com/cose-lalgoritmo-adaptive-boosting-adaboost/>.
- [11] <https://www.lorenzogovoni.com/funzione-di-perdita/>.

- [12] <https://www.lorenzogovoni.com/knn/>.
- [13] <https://www.lorenzogovoni.com/random-forest/>.
- [14] <https://www.lorenzogovoni.com/support-vector-machine/>.
- [15] <https://www.med4.care/curva-roc-receiver-operating-characteristic-introduzione-e-applicazione-ai-test-dagnostici/>.
- [16] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6. Ieee, 2017.
- [17] Thilini Ariyachandra and Hugh J Watson. Which data warehouse architecture is most successful? *Business intelligence journal*, 11(1):4, 2006.
- [18] Vercellis Carlos. Business intelligence systems: Data mining and optimization for decision making, 2009.
- [19] Bojan Cestnik and Stojan Košti. Constructing semantic ontologies for business analytics. In *Proceedings of the 11th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing on International Conference on Computer Systems and Technologies*, pages 436–441, 2010.
- [20] Cong Cheng, Huihui Zhong, and Liebing Cao. Facilitating speed of internationalization: The roles of business intelligence and organizational agility. *Journal of Business Research*, 110:95–103, 2020.
- [21] Dr Ossama Embarak, Embarak, and Karkal. *Data analysis and visualization using python*. Springer, 2018.
- [22] Vitaly Friedman. Data visualization and infographics. *Graphics, Monday Inspiration*, 14:2008, 2008.
- [23] Matt How. Beyond the modern data warehouse. In *The Modern Data Warehouse in Azure*, pages 229–274. Springer, 2020.

- [24] Matt How. *The Modern Data Warehouse in Azure: Building with Speed and Agility on Microsoft's Cloud Platform*. Springer, 2020.
- [25] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [26] Giovanni Montemurro. *Business Intelligence nell'era dei Big Data= Business Intelligence in the Big Data era*. PhD thesis, Politecnico di Torino, 2020.
- [27] Roger Moser. How (today) technology integration might matter more than technology innovation: A decision model innovation perspective. 2018.
- [28] Foster Provost and Tom Fawcett. Data science and its relationship to big data and data-driven decision making. *Big data*, 1(1):51–59, 2013.
- [29] Jayanthi Ranjan. Business intelligence: Concepts, components, techniques and benefits. *Journal of theoretical and applied information technology*, 9(1):60–70, 2009.
- [30] Marco Russo and Alberto Ferrari. *The Definitive Guide to DAX: Business intelligence for Microsoft Power BI, SQL Server Analysis Services, and Excel*. Microsoft Press, 2019.
- [31] David Reinsel-John Gantz-John Rydning. The digitization of the world from edge to core. *Framingham: International Data Corporation*, page 16, 2018.
- [32] Soraya Sedkaoui and Salim Moualdi. Big data analytics for the small social enterprise. *DATA ANALYTICS 2018*, page 29, 2018.
- [33] Rüdiger Wirth and Jochen Hipp. Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, volume 1. Springer-Verlag London, UK, 2000.

- [34] Zhen Xiao, Weijia Song, and Qi Chen. Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE transactions on parallel and distributed systems*, 24(6):1107–1117, 2012.
- [35] Yu-Jin Zhang. Image engineering. In *Handbook of Image Engineering*, pages 55–83. Springer, 2021.
- [36] Zhi-Hua Zhou and SF Chen. Neural network ensemble. *CHINESE JOURNAL OF COMPUTERS-CHINESE EDITION-*, 25(1):1–8, 2002.